FORTINET

API Reference

Lacework FortiCNAPP

FORTINET DOCUMENT LIBRARY
https://docs.fortinet.com

FORTINET VIDEO LIBRARY
https://video.fortinet.com

FORTINET BLOG
https://blog.fortinet.com

CUSTOMER SERVICE & SUPPORT
https://support.fortinet.com

FORTINET TRAINING & CERTIFICATION PROGRAM
https://www.fortinet.com/training-certification

FORTINET TRAINING INSTITUTE
https://training.fortinet.com

FORTIGUARD LABS
https://www.fortiguard.com

END USER LICENSE AGREEMENT
https://www.fortinet.com/doc/legal/EULA.pdf

FEEDBACK
Email: techdoc@fortinet.com

April 9, 2025
Lacework FortiCNAPP 25.1.0 API Reference
91-252-1083648-20250409

TABLE OF CONTENTS

About the Lacework FortiCNAPP API

With the Lacework FortiCNAPP API, you can access many features of the Lacework FortiCNAPP platform programmatically. For example, you can:

- Create alert channels, alert profiles, and report rules
- Search and scan container vulnerabilities
- Obtain audit logs
- Create cloud accounts and container registries
- Get information about compliance configurations
- Search for applications running on a machine
- Create a Lacework Query Language (LQL) policy for custom alerts

As a REST API, Lacework FortiCNAPP API works with common REST API tools, such as curl or Postman.

The Lacework FortiCNAPP API reference documentation provides a complete list of the Lacework FortiCNAPP API endpoints, along with schema descriptions and sample requests and responses.

You can access the Lacework FortiCNAPP API reference documentation from the Lacework FortiCNAPP Console. From the *Help* menu, select *API Documentation* or *API 2.0 Documentation*. You can also access the reference documentation directly at the following URL:

https://api.lacework.net/api/v2/docs

⚠ API v1 is no longer supported.

API Keys and Access Tokens

The Lacework FortiCNAPP API uses bearer authentication to authorize API requests. To use bearer tokens, create an API access key in the Lacework FortiCNAPP Console. You can then use the key ID and the generated secret to get temporary access tokens for API requests.

Create API Keys

To create API keys, you must have the account admin role or otherwise have write permissions for API keys in the Lacework FortiCNAPP Console. See Access Control Overview for more information.

API keys apply to the account level only; that is, keys do not apply at the org level, across multiple accounts. You need to create a key in each account in which you want to use the API.

An API key gives a user full access to the Lacework FortiCNAPP API. This level of permission allocation may not be desirable for all organizations, especially to those seeking to adhere to principles of least privilege. To more closely control API access, instead of granting keys to particular users, you can create a service user with read-only access, and assign user permissions for that user to the required API endpoints. For more information about service users, see Service Users.

Each user can have up to 20 API keys. An API key doesn't expire but it can be disabled or deleted.

Create an API key in the Lacework FortiCNAPP Console in the *Settings* > *Configuration* > *API keys* page. After creating a key, you should download it and store it securely.

For details on creating a key in the Lacework FortiCNAPP Console, see API Keys.

Temporary API Tokens

Once you have an API key, you can generate temporary API access (bearer) tokens to use to access the Lacework FortiCNAPP API. Use the Lacework FortiCNAPP API's POST api/v2/access/tokens operation to create temporary API access (bearer) tokens.

Method	POST
URL	https://YourLacework.lacework.net/api/v2/access/tokens
Headers	X-LW-UAKS:YourSecretKey Content-Type:application/json
Request Body	{ "keyId": YourAccessKeyID", "expiryTime": 3600 }

Replace YourSecretKey, YourAccessKeyID, and YourLacework with your values.

The `expiryTime` parameter is optional. If omitted from the request body, `expiryTime` defaults to 3600 seconds. The maximum `expiryTime` allowed is one day, 86400 seconds.

Generate Token Using Curl

To generate the API access token using curl, use a command in the following form:

```
curl -H "X-LW-UAKS:<YOUR_SECRET_KEY>" -H "Content-Type: application/json" -X POST -d
'{"keyId": "<YOUR_ACCESS_KEY_ID>", "expiryTime":3600}' https://<YOUR_LACEWORK_
URL>.lacework.net/api/v2/access/tokens
```

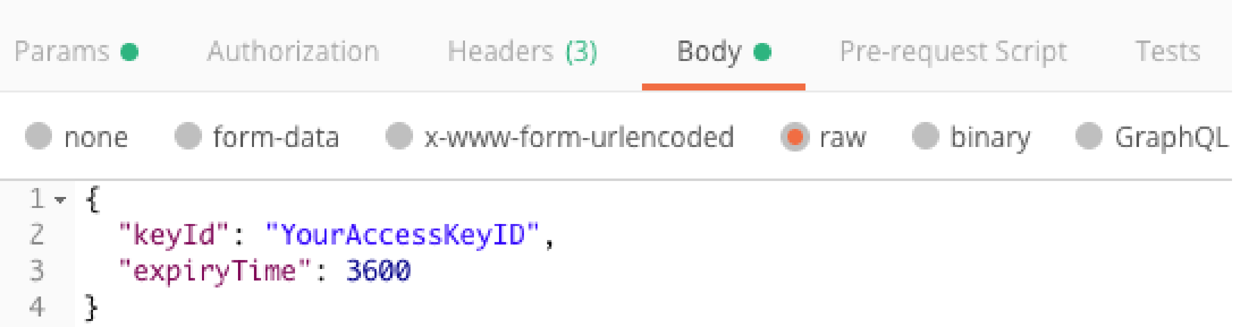
To set a expiry time (other than the default), specify the `expiryTime` value in the body of the request:

```
curl -H "X-LW-UAKS:<YOUR_SECRET_KEY>" -H "Content-Type: application/json" -X POST -d
'{"keyId": "<YOUR_ACCESS_KEY_ID>", "expiryTime":3600}' https://<YOUR_LACEWORK_
URL>.lacework.net/api/v2/access/tokens --data-raw '{ "keyId":"<YOUR_ACCESS_KEY_ID>",
"expiryTime": 86400 }'
```

Replace YOUR_SECRET_KEY, YOUR_ACCESS_KEY_ID, and YOUR_LACEWORK_URL with your values.

Generate Token Using Postman

To generate the API access token using Postman, construct your request as shown in the following image:



```
{
  "keyId": "YourAccessKeyID",
  "expiryTime": 3600
}
```

Response Body

The response body returns the token and token expiration time in the following form:

```
{
  “token”: “string”,
  “expiresAt”: “datetime”
}
```

Create Policies with the Lacework FortiCNAPP API

Creating custom policies lets you supplement the built-in Lacework FortiCNAPP policies with policies that are specialized for your requirements and environment. There are several ways to create policies in Lacework FortiCNAPP, including through the Lacework FortiCNAPP Console.

The following topics describe how to create policies with the Lacework FortiCNAPP API, from creating the LQL query to defining the policy that uses it. Before starting, make sure you have acquired an API key.

- Create a Policy with the API
- Create an Alert Profile

Custom Policy Types

There are several types of Lacework FortiCNAPP policies, violation, compliance, and manual.

You can create custom policies using the CLI or API. When creating a custom policy, you pass a policy definition as a JSON file. The policy types with example definitions appear below.

Lacework FortiCNAPP supports custom violation and compliance policies.

Violation

Violation policies check for activity violations. For example, checking violations from CloudTrail or Kubernetes audit log activity. These policies generate one alert for each violation.

Violation policies contain an `alertProfile` field, which controls the information that is surfaced for an alert (the "5 Ws" in the Lacework FortiCNAPP Console seen in, for example, *Resources* > *Cloud* > *AWS CloudTrail*.

```
{
  "policyId": "lacework-global-1",
  "title": "Virtual Private Cloud (VPC) Change",
  "enabled": true,
  "policyType": "Violation",
  "alertEnabled": true,
  "alertProfile": "LW_CloudTrail_Alerts.VPCChange_AwsResource",
  "evalFrequency": "Hourly",
  "queryId": "LW_Global_AWS_CTA_VPCChange",
  "severity": "medium",
  "description": "A VPC was created, deleted or changed",
  "remediation": "Check that the VPC change was expected.\nEnsure only specified
users can modify VPCs."
}
```

Compliance

Compliance policies check for configuration compliance, such as whether AWS resources are properly configured. Every resource that violates a policy can have multiple reasons for non-compliance. In contrast with the violation policies, these compliance policies generate one alert per policy. For example, if three S3 buckets violate a policy, Lacework FortiCNAPP generates only one alert that lists the non-compliant resources.

```
{
  "policyId": "lacework-global-75",
  "title": "Ensure CloudTrail log file validation is enabled",
  "enabled": false,
  "policyType": "Compliance",
  "alertEnabled": false,
  "severity": "low",
  "description": "CloudTrail log file validation creates a digitally signed
digest\nfile containing a hash of each log that CloudTrail writes to S3. These
digest\nfiles can be used to determine whether a log file was changed, deleted, or
unchanged\nafter CloudTrail delivered the log. It is recommended that file validation
be\nenabled on all CloudTrails.",
  "remediation": "Perform the following to enable log file validation on a given
trail:\nFrom Console:\n1. Sign in to the AWS Management Console and open the IAM
console at (https://console.aws.amazon.com/cloudtrail)\n2. Click on Trails on the
left navigation pane\n3. Click on target trail\n4. Within the S3 section click on the
edit icon (pencil)\n5. Click Advanced\n6. Click on the Yes radio button in section
Enable log file validation\n7. Click Save\nFrom Command Line:\naws cloudtrail update-
trail --name <trail_name> --enable-log-file-validation\nNote that periodic validation
of logs using these digests can be performed by running the following command:\naws
cloudtrail validate-logs --trail-arn <trail_arn> --start-time <start_time> --end-time
<end_time>",
  "references": [
    "CCE-78914-9",
    "https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-log-file-
validation-enabling.html"
  ],
  "infoLink": "https://docs.lacework.com/compliance/policies/lacework-global-75",
  "queryId": "LW_Global_AWS_Config_CloudTrailLogFileValidationNotEnabled",
  "tags": [
    "security:compliance",
    "framework:cis-aws-1-4-0",
    "control:3.2",
    "identifier:CCE-78914-9",
    "domain:AWS",
    "subdomain:Configuration"
  ]
}
```

Manual

Custom manual policies are not supported.

Manual policies represent recommendations that are not able to be automated but still have information to provide and are included in compliance reports. However, manual policies do not result in an automated assessment.

```
{
  "policyId": "lacework-global-32",
  "title": "Register security contact information",
  "policyType": "Manual",
  "severity": "low",
  "description": "AWS provides customers with the option of specifying the
contact\ninformation for account's security team. It is recommended that this
information\nbe provided.",
  "remediation": "Perform the following to establish security contact
information:\nFrom Console:\n1. Click on your account name at the top right corner of
the console.\n2. From the drop-down menu Click My Account\n3. Scroll down to the
Alternate Contacts section\n4. Enter contact information in the Security
section\nNote: Consider specifying an internal email distribution list to ensure
emails\nare regularly monitored by more than one individual.",
  "references": [
    "CCE-79200-2"
  ],
  "infoLink": "https://docs.lacework.com/compliance/policies/lacework-global-32",
  "tags": [
    "security:compliance",
    "framework:cis-aws-1-4-0",
    "control:1.2",
    "identifier:CCE-79200-2",
    "domain:AWS",
    "subdomain:Configuration"
  ]
}
```

Creating a Policy with the API

This topic walks you through using the Lacework FortiCNAPP API and Postman to create a custom LQL-based policy that checks for unrestricted ingress to TCP port 445.

If you are new to the Lacework FortiCNAPP API, see Get Started.

Policy Query Definition

What Datasources Are Available

The easiest way to learn about Lacework Query Language (LQL) datasources is to discover the names of the datasources and then get details about the one you are interested in.

To list all datasources that you can write a query against, use this endpoint:

```
GET https://AccountName.lacework.net/api/v2/Datasources
```

What Fields Can I Use from a Datasource

The examples use the following datasources:

- AWS datasource: `LW_CFG_AWS_EC2_SECURITY_GROUPS`
- Google Cloud datasource (Google Cloud datasources are currently in beta): `LW_CFG_GCP_COMPUTE_FIREWALL`
- Azure datasource (Azure datasources are currently in beta): `LW_CFG_AZURE_NETWORK_NETWORKSECURITYGROUPS`

Use the `GET https://AccountName.lacework.net/api/v2/Datasources/{datasource}` endpoint to examine the specified datasource's fields:

```
GET https://AccountName.lacework.net/api/v2/Datasources/LW_CFG_AWS_EC2_SECURITY_GROUPS
```

```
GET https://AccountName.lacework.net/api/v2/Datasources/LW_CFG_GCP_COMPUTE_FIREWALL
```

```
GET https://AccountName.lacework.net/api/v2/Datasources/LW_CFG_AZURE_NETWORK_NETWORKSECURITYGROUPS
```

Explore Datasources Using LQL

The `POST https://AccountName.lacework.net/api/v2/Queries/execute` endpoint executes a query on demand. On-demand execution differs from execution by ID in that the query you want to execute does not have to already exist in your Lacework FortiCNAPP instance. On-demand execution is useful for exploring datasources.

This example explores the `LW_CFG_AWS_EC2_SECURITY_GROUPS` datasource. Replace the datasource with `LW_CFG_GCP_COMPUTE_FIREWALL` or `LW_CFG_AZURE_NETWORK_NETWORKSECURITYGROUPS` respectively if using Google Cloud or Azure.

Example Query

The following is example `queryText`, which you will use with the endpoint:

```
{
   source {
      LW_CFG_AWS_EC2_SECURITY_GROUPS
   }
   return {
      RESOURCE_CONFIG
   }
}
```

Format the Query

To use the query with the endpoint, you must remove all line breaks.

```
"queryText": "{source {LW_CFG_AWS_EC2_SECURITY_GROUPS} return {RESOURCE_CONFIG}}"
```

Execute the Query

To execute the query, use this endpoint:

```
POST https://AccountName.lacework.net/api/v2/Queries/execute
```

In the body input parameter, pass in the following:

- `queryText`
- LQL query arguments (`StartTimeRange` and `EndTimeRange`)

The request body would be similar to the following:

```
{
    "query": {
        "queryText": "{source {LW_CFG_AWS_EC2_SECURITY_GROUPS} return {RESOURCE_
CONFIG}}"
    },
    "arguments": [
        {"name": "StartTimeRange", "value": "2022-07-20T00:00:00.000Z"},
        {"name": "EndTimeRange", "value": "2022-07-21T00:00:00.000Z"}
    ]
}
```

```
{
 "RESOURCE_CONFIG": {
   "Description": "default VPC security group",
   "GroupId": "sg-000",
   "GroupName": "default",
   "IpPermissions": [
     {
       "IpProtocol": "-1",
       "IpRanges": [],
       "Ipv6Ranges": [],
       "PrefixListIds": [],
       "UserIdGroupPairs": [
         {
           "GroupId": "sg-000",
           "UserId": "111"
         }
       ]
     }
   ],
   "IpPermissionsEgress": [
     {
       "IpProtocol": "-1",
       "IpRanges": [
         {
           "CidrIp": "0.0.0.0/0"
         }
       ],
       "Ipv6Ranges": [],
       "PrefixListIds": [],
       "UserIdGroupPairs": []
     }
   ],
   "OwnerId": "111",
   "VpcId": "vpc-000"
 }
}
```

Create a Query

This query checks security groups for unrestricted ingress to TCP port 445.

Example Query

Use the example `queryText` that corresponds to your cloud provider.

```
{
   source {
       LW_CFG_AWS_EC2_SECURITY_GROUPS a,
       array_to_rows(a.RESOURCE_CONFIG:IpPermissions) as (ip_permissions),
       array_to_rows(ip_permissions:IpRanges) as (ip_ranges)
   }
   filter {
       ip_permissions:IpProtocol = 'tcp'
       and ip_permissions:FromPort = 445
       and ip_permissions:ToPort = 445
       and ip_ranges:CidrIp = '0.0.0.0/0'
   }
   return distinct {
       ACCOUNT_ALIAS,
       ACCOUNT_ID,
       ARN as RESOURCE_KEY,
       RESOURCE_REGION,
       RESOURCE_TYPE,
       SERVICE
   }
}
```

```
{
   source {
       LW_CFG_GCP_COMPUTE_FIREWALL firewall,
       array_to_rows(firewall.RESOURCE_CONFIG:allowed) as (allowed),
       array_to_rows(allowed:ports) as (ports),
       array_to_rows(firewall.RESOURCE_CONFIG:sourceRanges) as (ranges)
   }
   filter {
       RESOURCE_CONFIG:direction = 'INGRESS'
       and allowed:IPProtocol = 'tcp'
       and ports = '445'
       and ranges = '0.0.0.0/0'
   }
   return distinct {
       ORGANIZATION_ID,
       PROJECT_NUMBER,
       PROJECT_ID,
       FOLDER_IDS,
       URN as RESOURCE_KEY,
       RESOURCE_REGION,
       RESOURCE_TYPE,
       SERVICE
    }
}
```

```
{
   source {
       LW_CFG_AZURE_NETWORK_NETWORKSECURITYGROUPS a,
       array_to_rows(a.RESOURCE_CONFIG:securityRules) as (rules)
   }
   filter {
       rules:"properties".access = 'Allow'
       and rules:"properties".direction = 'Inbound'
       and rules:"properties".protocol = 'Tcp'
       and rules:"properties".destinationPortRange = '445'
       and rules:"properties".sourceAddressPrefix = '*'
   }
   return distinct {
       TENANT_ID,
       TENANT_NAME,
       SUBSCRIPTION_ID,
       SUBSCRIPTION_NAME,
       URN as RESOURCE_KEY,
       RESOURCE_REGION,
       RESOURCE_TYPE
    }
}
```

Format the Query

To use the query with the endpoint, you must remove all line breaks. Use the example that corresponds to your cloud provider.

Add this custom `queryId` for the example: `LW_Custom_UnrestrictedIngressToTCP445`.

```
{
  "queryText": "{source {LW_CFG_AWS_EC2_SECURITY_GROUPS a, array_to_rows(a.RESOURCE_
CONFIG:IpPermissions) as (ip_permissions), array_to_rows(ip_permissions:IpRanges) as
(ip_ranges)} filter {ip_permissions:IpProtocol = 'tcp' and ip_permissions:FromPort =
445 and ip_permissions:ToPort = 445 and ip_ranges:CidrIp = '0.0.0.0/0'} return
distinct {ACCOUNT_ALIAS, ACCOUNT_ID, ARN as RESOURCE_KEY, RESOURCE_REGION, RESOURCE_
TYPE, SERVICE}}",
  "queryId": "LW_Custom_UnrestrictedIngressToTCP445"
}
```

```
{
  "queryText": "{source {LW_CFG_GCP_COMPUTE_FIREWALL firewall, array_to_rows
(firewall.RESOURCE_CONFIG:allowed) as (allowed), array_to_rows(allowed:ports) as
(ports), array_to_rows(firewall.RESOURCE_CONFIG:sourceRanges) as (ranges)} filter
{RESOURCE_CONFIG:direction = 'INGRESS' and allowed:IPProtocol = 'tcp' and ports =
'445' and ranges = '0.0.0.0/0'} return distinct {ORGANIZATION_ID, PROJECT_NUMBER,
PROJECT_ID, FOLDER_IDS, URN as RESOURCE_KEY, RESOURCE_REGION, RESOURCE_TYPE,
SERVICE}}",
  "queryId": "LW_Custom_UnrestrictedIngressToTCP445"
}
```

```
{
  "queryText": "{source {LW_CFG_AZURE_NETWORK_NETWORKSECURITYGROUPS a, array_to_rows
(a.RESOURCE_CONFIG:securityRules) as (rules)} filter {rules:"properties".access =
'Allow' and rules:"properties".direction = 'Inbound' and rules:"properties".protocol
= 'Tcp' and rules:"properties".destinationPortRange = '445' and
rules:"properties".sourceAddressPrefix = '*'} return distinct {TENANT_ID, TENANT_
NAME, SUBSCRIPTION_ID, SUBSCRIPTION_NAME, URN as RESOURCE_KEY, RESOURCE_REGION,
RESOURCE_TYPE}}",
  "queryId": "LW_Custom_UnrestrictedIngressToTCP445"
}
```

Use your query with the `POST https://AccountName.lacework.net/api/v2/Queries` endpoint as discussed in POST Queries.

Create a Policy

Create a policy that uses your new query. Use the example that corresponds to your cloud provider.

```
{
  "title": "Security Groups Should Not Allow Unrestricted Ingress to TCP Port 445",
  "enabled": true,
  "policyType": "Violation",
  "alertEnabled": true,
  "alertProfile": "LW_CFG_AWS_DEFAULT_PROFILE.CFG_AWS_Violation",
  "evalFrequency": "Daily",
  "queryId": "LW_Custom_UnrestrictedIngressToTCP445",
  "severity": "high",
  "description": "Security groups should not allow unrestricted ingress to TCP port
445",
  "remediation": "Policy remediation"
}
```

```
{
  "title": "Security Groups Should Not Allow Unrestricted Ingress to TCP Port 445",
  "enabled": true,
  "policyType": "Violation",
  "alertEnabled": true,
  "alertProfile": "LW_CFG_GCP_DEFAULT_PROFILE.Violation",
  "evalFrequency": "Daily",
  "queryId": "LW_Custom_UnrestrictedIngressToTCP445",
  "severity": "high",
  "description": "Security groups should not allow unrestricted ingress to TCP port
445",
  "remediation": "Policy remediation"
}
```

```
{
  "title": "Network Security Groups Should Not Allow Unrestricted Ingress to TCP Port
445",
  "enabled": true,
  "policyType": "Violation",
  "alertEnabled": true,
  "alertProfile": "LW_CFG_AZURE_DEFAULT_PROFILE.Violation",
  "evalFrequency": "Daily",
  "queryId": "LW_Custom_UnrestrictedIngressToTCP445",
  "severity": "high",
  "description": "Network security groups should not allow unrestricted ingress to
TCP port 445",
  "remediation": "Policy remediation"
}
```

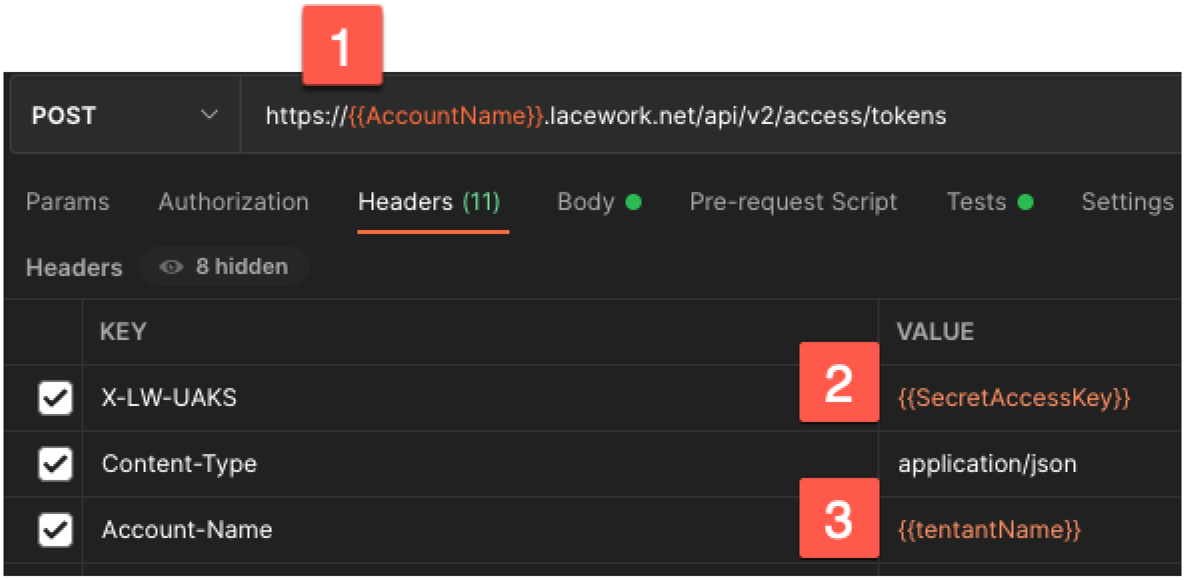
Use your policy with the `POST https://AccountName.lacework.net/api/v2/Policies` endpoint as discussed in POST Policies.

Postman Configuration

Request a Bearer Token Using POST

Headers

```
https://AccountName.lacework.net/api/v2/access/tokens
```



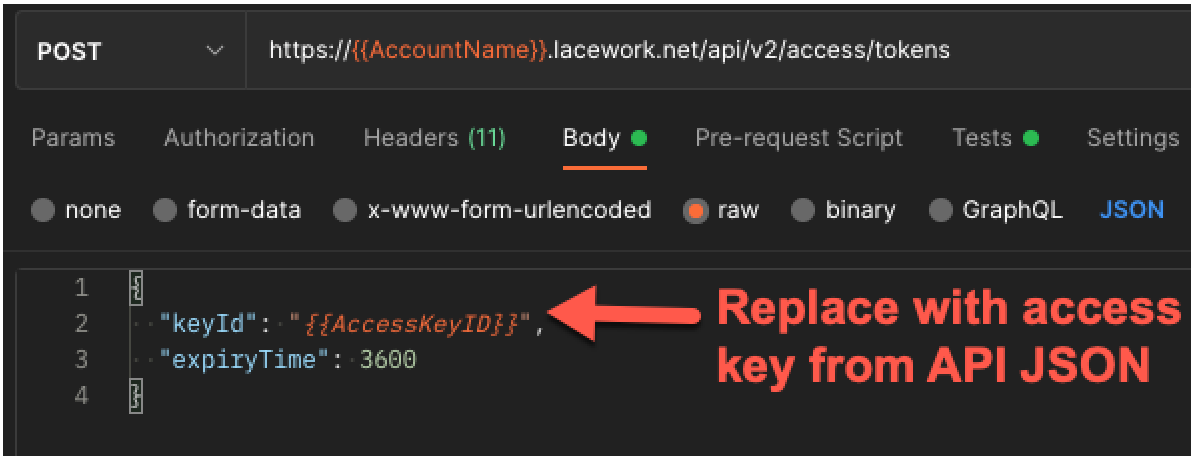
POST https://{{AccountName}}.lacework.net/api/v2/access/tokens

	KEY	VALUE
☑	X-LW-UAKS	{{SecretAccessKey}}
☑	Content-Type	application/json
☑	Account-Name	{{tentantName}}

1. Replace with your Lacework FortiCNAPP account name.
2. Replace with the secret from API JSON.
3. If the account is an org, set this if applying to the tenant. Deselect if not an org.

Body

```
{
  "keyId": "AccessKeyID",
  "expiryTime": 3600
}
```



POST https://{{AccountName}}.lacework.net/api/v2/access/tokens

```
{
  "keyId": "{{AccessKeyID}}",
  "expiryTime": 3600
}
```

Replace with access key from API JSON

Replace with the access key from the API JSON.

Set a Token Variable

Optionally, you can use Postman's scripting capabilities to update the authentication token variable automatically after generating a bearer token. Your script should retrieve the JSON response body and parse out the required JSON object value and pass it to the collection variable.

If you do not configure Postman to automatically update the authentication token variable, you can update the token manually for each endpoint.

GET Datasources

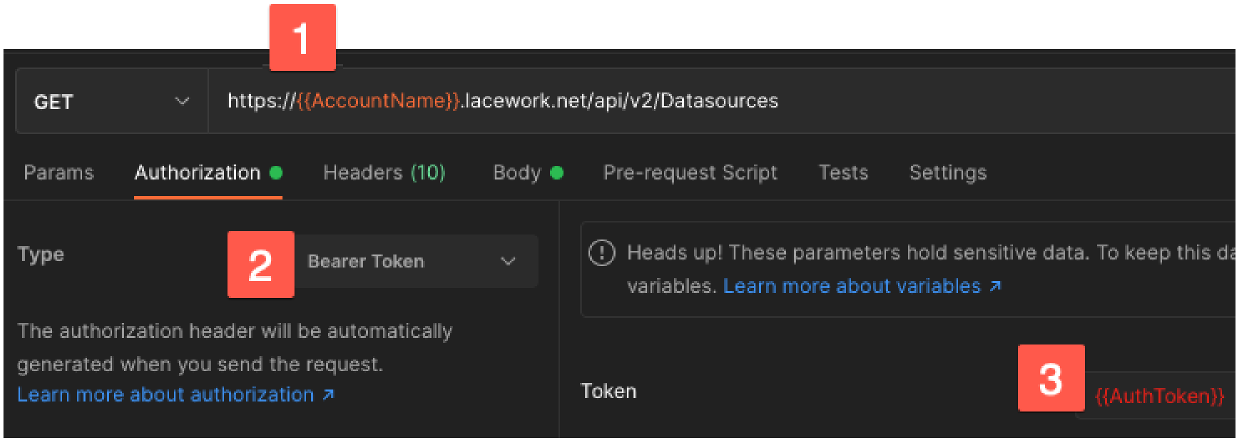
To list all AWS datasources, use this endpoint:

`https://AccountName.lacework.net/api/v2/Datasources`

This example uses the `LW_CFG_AWS_EC2_SECURITY_GROUPS` datasource. To examine its details only, use this endpoint:

`https://AccountName.lacework.net/api/v2/Datasources/LW_CFG_AWS_EC2_SECURITY_GROUPS`

Authorization



GET https://{{AccountName}}.lacework.net/api/v2/Datasources

Type Bearer Token

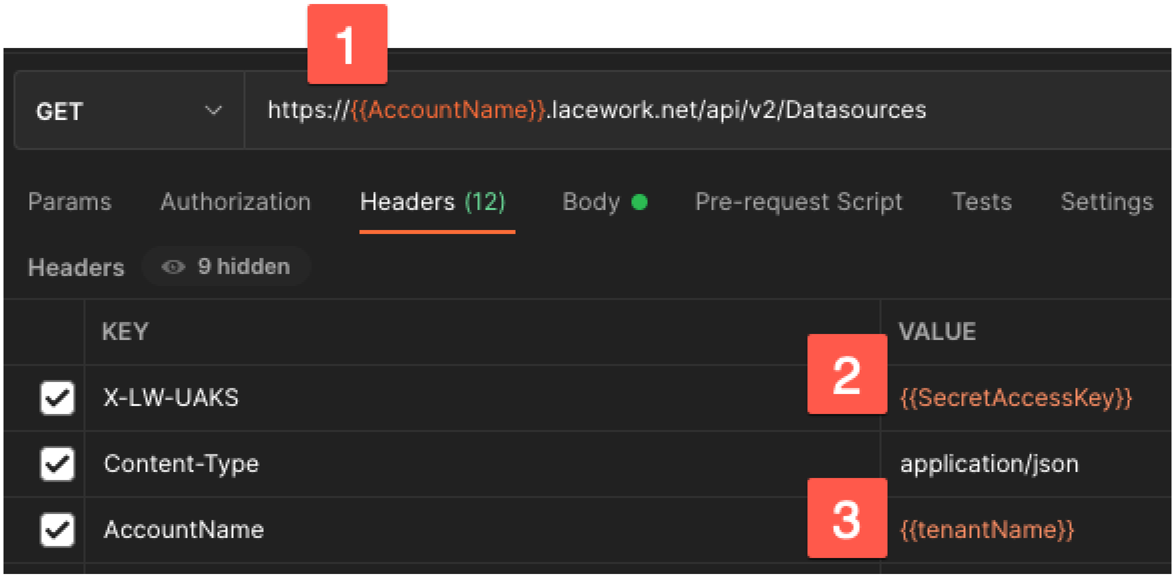
The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this da
variables. Learn more about variables ↗

Token {{AuthToken}}

1. Replace with your Lacework FortiCNAPP account name.
2. Select Bearer Token.
3. Replace with the token returned from the request POST.

Headers

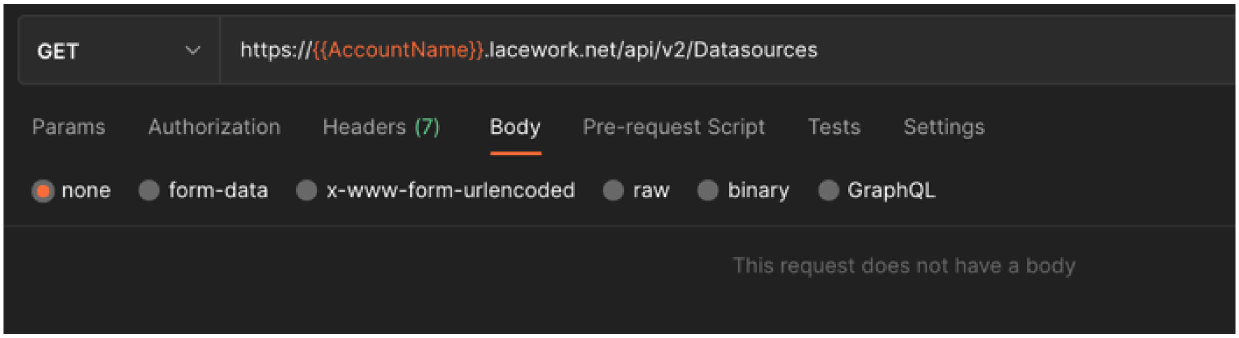


GET https://{{AccountName}}.lacework.net/api/v2/Datasources

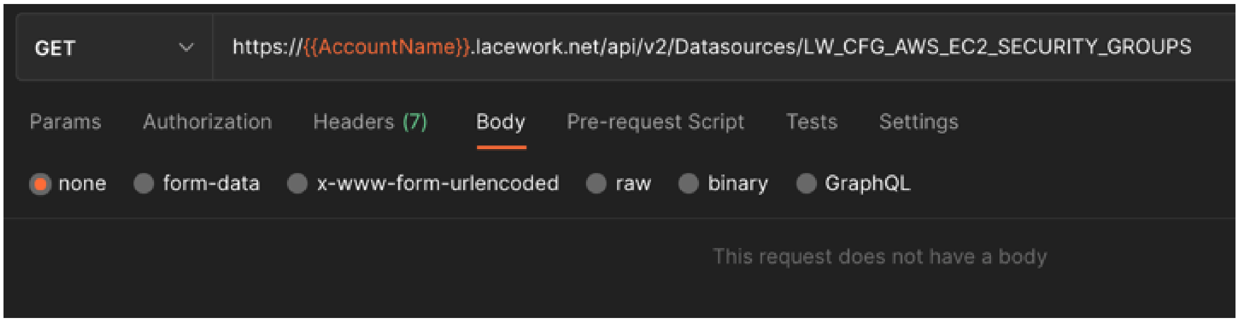
	KEY	VALUE
☑	X-LW-UAKS	{{SecretAccessKey}}
☑	Content-Type	application/json
☑	AccountName	{{tenantName}}

1. Replace with your Lacework FortiCNAPP account name.
2. Replace with the secret from API JSON.
3. If the account is an org, set this if applying to the tenant. Deselect if not an org.

Endpoints


GET https://{{AccountName}}.lacework.net/api/v2/Datasources

This request does not have a body


GET https://{{AccountName}}.lacework.net/api/v2/Datasources/LW_CFG_AWS_EC2_SECURITY_GROUPS

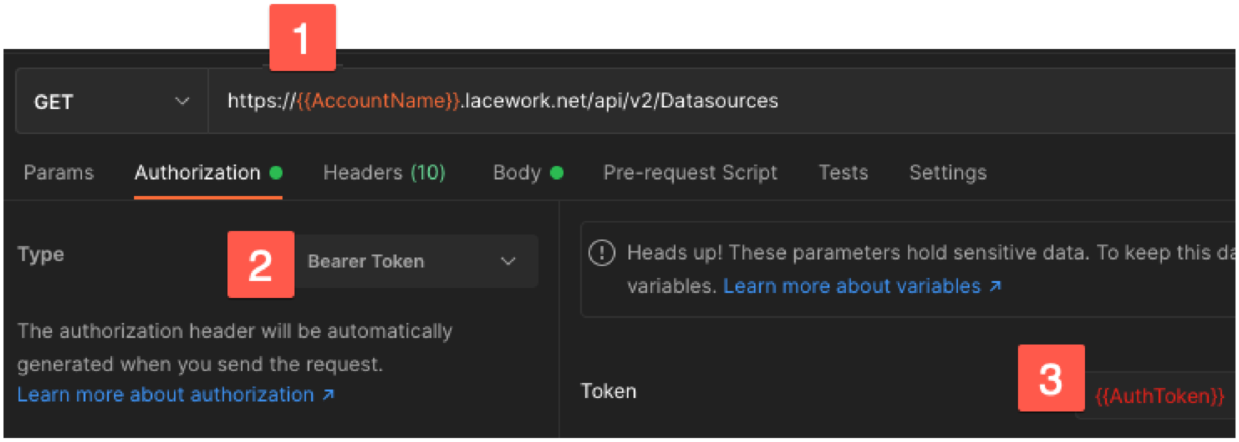
This request does not have a body

POST Queries

To add a query to your Lacework FortiCNAPP instance, use this endpoint.

```
https://AccountName.lacework.net/api/v2/Queries
```

Authorization


GET https://{{AccountName}}.lacework.net/api/v2/Datasources

Type Bearer Token

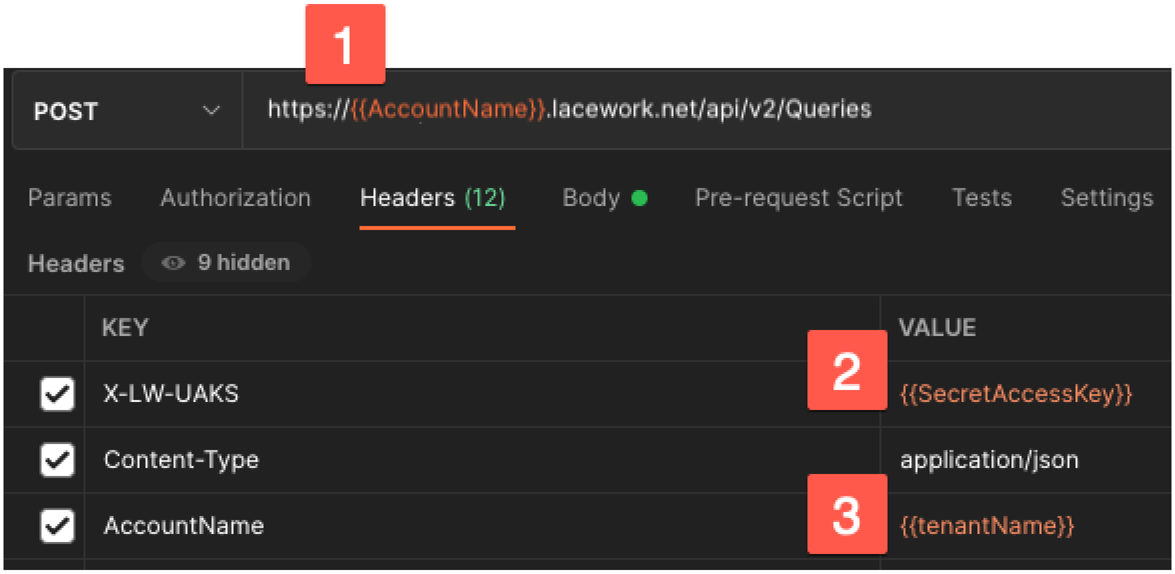
The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this da
variables. Learn more about variables ↗

Token {{AuthToken}}

1. Replace with your Lacework FortiCNAPP account name.
2. Select Bearer Token.
3. Replace with the token returned from the request POST.

Headers

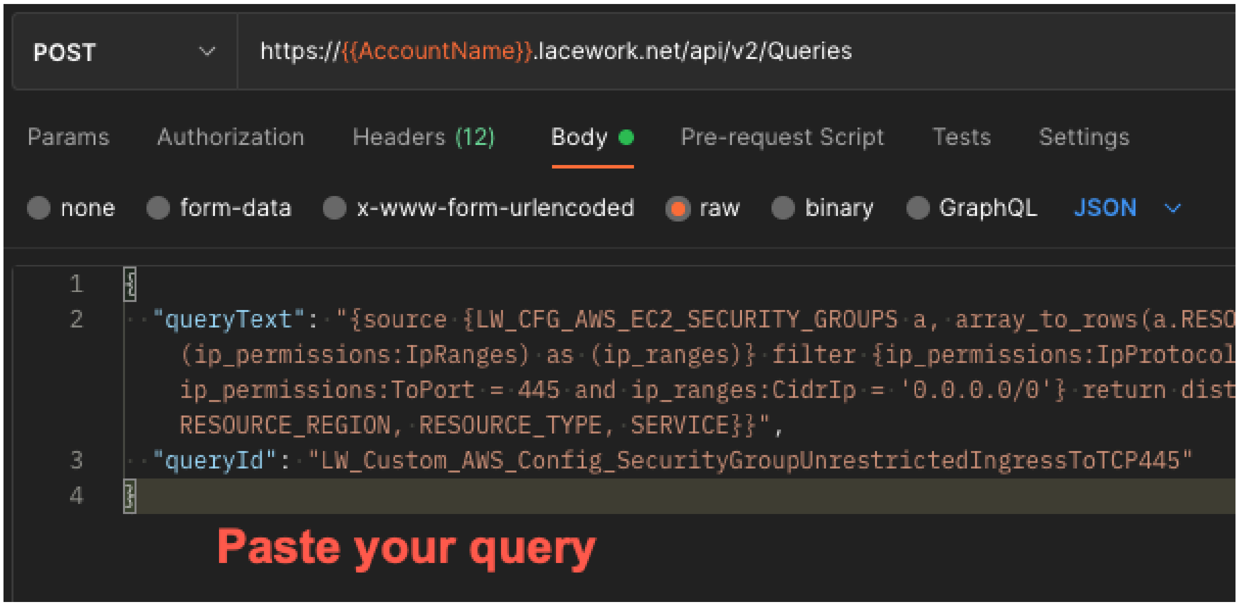


POST https://{{AccountName}}.lacework.net/api/v2/Queries

	KEY	VALUE
☑	X-LW-UAKS	{{SecretAccessKey}}
☑	Content-Type	application/json
☑	AccountName	{{tenantName}}

1. Replace with your Lacework FortiCNAPP account name.
2. Replace with the secret from API JSON.
3. If the account is an org, set this if applying to the tenant. Deselect if not an org.

Body



POST https://{{AccountName}}.lacework.net/api/v2/Queries

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
  "queryText": "{source {LW_CFG_AWS_EC2_SECURITY_GROUPS a, array_to_rows(a.RESO
    (ip_permissions:IpRanges) as (ip_ranges)} filter {ip_permissions:IpProtocol
    ip_permissions:ToPort = 445 and ip_ranges:CidrIp = '0.0.0.0/0'} return dist
    RESOURCE_REGION, RESOURCE_TYPE, SERVICE}}",
  "queryId": "LW_Custom_AWS_Config_SecurityGroupUnrestrictedIngressToTCP445"
}
```

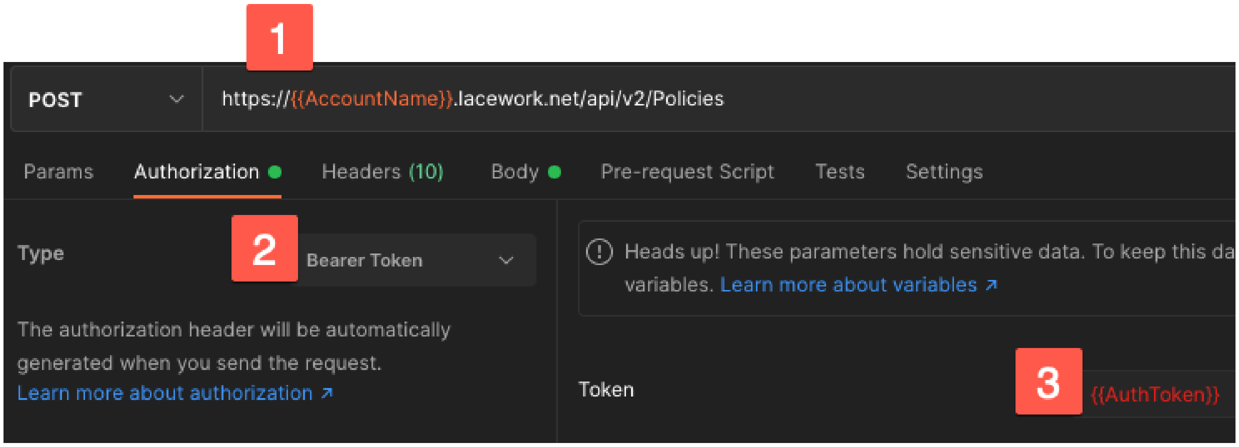
Paste your query into the request body.

POST Policies

To add a policy to your Lacework FortiCNAPP instance, use this endpoint.

```
https://AccountName.lacework.net/api/v2/Policies
```

Authorization



POST https://{{AccountName}}.lacework.net/api/v2/Policies

Type Bearer Token

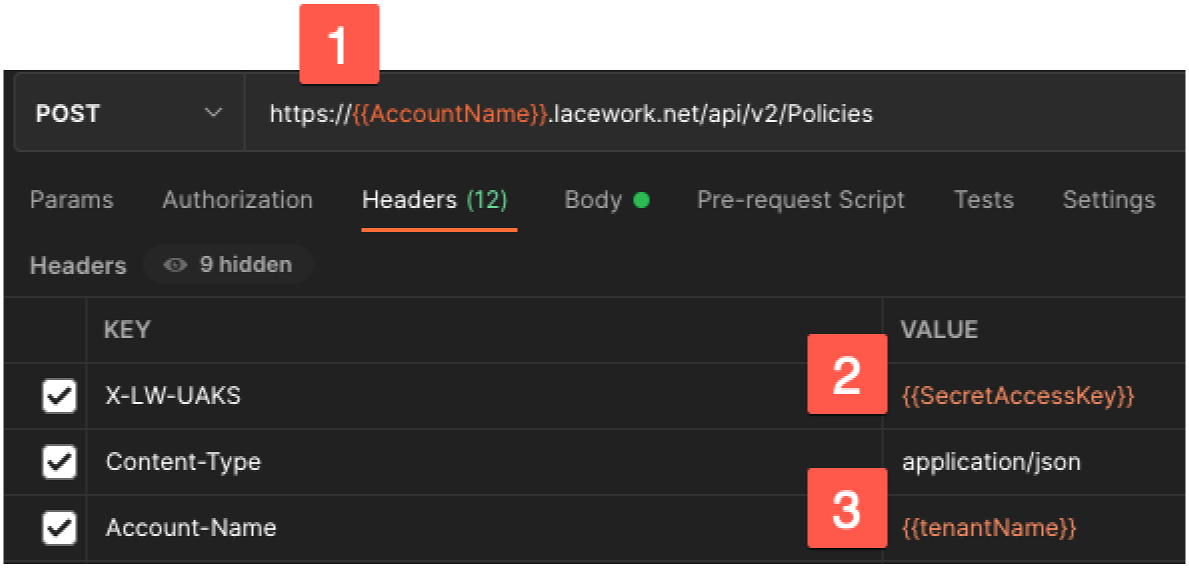
The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this da variables. Learn more about variables ↗

Token {{AuthToken}}

1. Replace with your Lacework FortiCNAPP account name.
2. Select Bearer Token.
3. Replace with the token returned from the request POST.

Headers



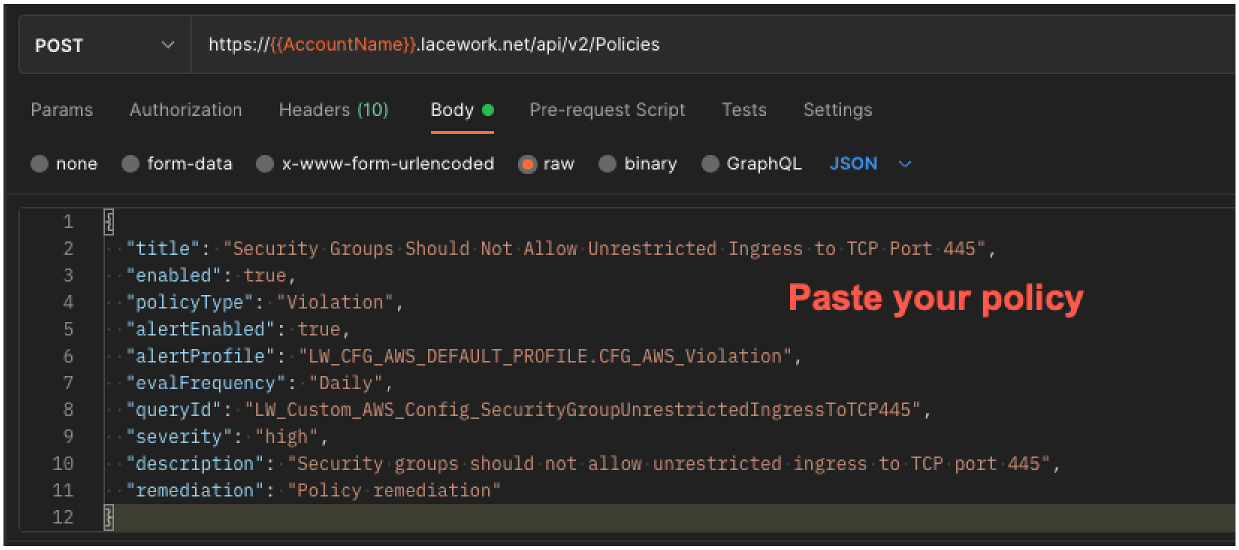
POST https://{{AccountName}}.lacework.net/api/v2/Policies

Headers 9 hidden

	KEY	VALUE
☑	X-LW-UAKS	{{SecretAccessKey}}
☑	Content-Type	application/json
☑	Account-Name	{{tenantName}}

1. Replace with your Lacework FortiCNAPP account name.
2. Replace with the secret from API JSON.
3. If the account is an org, set this if applying to the tenant. Deselect if not an org.

Body



```
{
  "title": "Security Groups Should Not Allow Unrestricted Ingress to TCP Port 445",
  "enabled": true,
  "policyType": "Violation",
  "alertEnabled": true,
  "alertProfile": "LW_CFG_AWS_DEFAULT_PROFILE.CFG_AWS_Violation",
  "evalFrequency": "Daily",
  "queryId": "LW_Custom_AWS_Config_SecurityGroupUnrestrictedIngressToTCP445",
  "severity": "high",
  "description": "Security groups should not allow unrestricted ingress to TCP port 445",
  "remediation": "Policy remediation"
}
```

Paste your policy

Paste your policy into the request body.

Create an Alert Profile

This topic describes how to create a custom alert profile that extends a predefined alert profile.

For example, you would use a custom alert profile when you want to customize the event's text on the Events page under the timeline, in the event summary, or the event's individual dossier.

All Lacework FortiCNAPP endpoints require an API access (bearer) token to be specified when you invoke the endpoint. If you already have a secret key, you can use the `POST /api/v2/access/tokens` endpoint to generate an access token. For details, see API Access Keys and Tokens.

Identify the Alert Profile to Extend

Because an alert profile helps to map what data is available for the alert message, each alert profile corresponds to a datasource. For example, if your query uses the `LW_CFG_AWS_EC2_SECURITY_GROUPS` datasource, extend the `LW_CFG_AWS_DEFAULT_PROFILE` alert profile. See Identifying Which Alert Profile to Use for more information.

This example shows the predefined `LW_CFG_AWS_DEFAULT_PROFILE` alert profile. This is the alert profile you will extend.

```
GET https://AccountName.lacework.net/api/v2/AlertProfiles/LW_CFG_AWS_DEFAULT_PROFILE
```

```
{
    "data": {
        "alertProfileId": "LW_CFG_AWS_DEFAULT_PROFILE",
        "extends": "LW_LPP_BaseProfile",
        "fields": [
            {
                "name": "_PRIMARY_TAG"
            },
            {
                "name": "RESOURCE_ID"
            }, ...
        ],
        "descriptionKeys": [
            {
                "name": "_OCCURRENCE",
                "spec": "{{_OCCURRENCE}}"
            },
            {
                "name": "RESOURCE_ID",
                "spec": "{{RESOURCE_ID}}"
            }, ...
        ],
        "alerts": [
            {
                "name": "CFG_AWS_NewViolation",
                "eventName": "LW Configuration AWS Violation Alert",
                "description": "{{_OCCURRENCE}} Violation for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}",
                "subject": "{{_OCCURRENCE}} violation detected for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}"
            },
            {
                "name": "CFG_AWS_PolicyChanged",
                "eventName": "LW Configuration AWS Violation Alert",
                "description": "{{_OCCURRENCE}} Violation for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}",
                "subject": "{{_OCCURRENCE}} violation detected for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}"
            },
            {
                "name": "CFG_AWS_Violation",
                "eventName": "LW Configuration AWS Violation Alert",
                "description": "{{_OCCURRENCE}} Violation for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}",
                "subject": "{{_OCCURRENCE}} violation detected for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}"
            }
        ]
    }
}
```

Customize the Alert Profile

From the alert profile that you want to extend, you only need its alert profile ID. Your custom alert profile inherits all of its other data.

Configure the custom alert profile using the following guidelines:

- `alertProfileId`: Specify a unique identification for the alert profile.
- `extends`: Specify the alert profile ID of the alert profile you want to extend. Don't extend `LW_LPP_BaseProfile` because it doesn't have alerts.

Under the `alerts` field:

- `name`: Specify a unique name for the alert template. If you specify an existing name, it overrides the existing alert template.
- `eventName`: Specify a meaningful name for the event. The `eventName` displays in the event summary and the event's individual dossier.
- `description`: Specify a description. You can use regular text and the available `descriptionKeys`. The `description` displays in the event summary and the event's individual dossier.
- `subject`: Specify a subject line. You can use regular text and the available `descriptionKeys`. The `subject` displays on the Events page under the timeline.

To see which description keys you can use in the `description` and `subject`, review the alert profile's data in the `GET /api/v2/AlertProfiles/{alertProfileId}` response.

Create the Alert Profile

To create the custom alert profile in your Lacework FortiCNAPP instance, call the `POST /api/v2/AlertProfiles` endpoint with the alert profile in the body, for example:

```
{
    "alertProfileId": "Custom_CFG_AWS_Profile",
    "extends": "LW_CFG_AWS_DEFAULT_PROFILE",
    "alerts": [
        {
            "name": "Custom_Violation",
            "eventName": "Custom Violation Alert",
            "description": "Violation for AWS Resource {{RESOURCE_TYPE}}:{{RESOURCE_
ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_REGION}}",
            "subject": "Violation detected for AWS Resource {{RESOURCE_TYPE}}:
{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_REGION}}"
        }
    ]
}
```

If successful, it returns a response:

```
{
    "alertProfileId": "Custom_CFG_AWS_Profile",
    "extends": "LW_CFG_AWS_DEFAULT_PROFILE",
    ...
}
```

The returned response includes the data from your custom alert profile and all data inherited from its parent alert profile.

Modify an Alert Template within an Alert Profile

You can modify a specified alert template within an alert profile. Use the endpoints described in the following sections.

Create an Alert Template

Use the `POST /api/v2/AlertProfiles/{alertProfileId}/AlertTemplates` endpoint to create an alert within an existing custom alert profile.

For example, to create another alert template in the example profile from Create the Alert Profile, call this endpoint:

```
POST https://AccountName.lacework.net/api/v2/AlertProfiles/Custom_CFG_AWS_
Profile/AlertTemplates
```

Provide the alert template fields in the request body:

```
{
    "name": "Another_Custom_Violation",
    "eventName": "Another Custom Violation Alert",
    "description": "Violation for AWS Resource {{RESOURCE_TYPE}}:{{RESOURCE_ID}} in
account {{ACCOUNT_ID}} region {{RESOURCE_REGION}}",
     "subject": "Violation detected for AWS Resource {{RESOURCE_TYPE}}:{{RESOURCE_
ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_REGION}}"
}
```

Update an Alert Template

Use the `PATCH /api/v2/AlertProfiles/{alertProfileId}/AlertTemplates/{alertTemplateName}` endpoint to update an alert template within a custom alert profile.

For example, to update the example alert template from Create the Alert Profile, call this endpoint:

```
PATCH https://AccountName.lacework.net/api/v2/AlertProfiles/Custom_CFG_AWS_
Profile/AlertTemplates/Custom_Violation
```

Provide the updated alert template fields in the request body (all fields are optional):

```
{
    "eventName": "Revised Event Name for the Alert",
    "description": "Revised description",
    "subject": "Revised subject for violation detected"
}
```

Delete an Alert Template

Use the `DELETE /api/v2/AlertProfiles/{alertProfileId}/AlertTemplates/{alertTemplateName}` endpoint to delete an alert template from a custom alert profile.

For example, to delete the example alert template from Create the Alert Profile, call this endpoint:

```
DELETE https://AccountName.lacework.net/api/v2/AlertProfiles/Custom_CFG_AWS_
Profile/AlertTemplates/Custom_Violation
```

Next Steps

Create a new policy or update an existing policy to use the alert profile. In the policy's `alertProfile` field, specify the alert profile ID and the alert template name in this format: `alertProfileId.alertTemplateName`.

To use the example `Custom_CFG_AWS_Profile` alert profile and alert template above, you would specify the following in the `alertProfile` field: `Custom_CFG_AWS_Profile.Custom_Violation`.

Bulk Policy Update

BETA FEATURE This article describes functionality that is currently in beta.

The Lacework FortiCNAPP API enables you to create, delete, and update Lacework FortiCNAPP policies programmatically. You can update policies individually, using the Update Policies endpoint, or multiple policies at once, using the Bulk Update Policies (beta) endpoint described here.

Bulk Update Guidelines

The Bulk Update Policy API lets you change the status (enabled or disabled) or severity of multiple policies at a time.

The API reference documentation provides overview and usage information for the API. However, there are additional guidelines applicable to the use of the Bulk Update Policy API, as follows.

General Guidelines

General guidelines for using the Bulk Update Policy API include:

1. Bulk operations to enable/disable policies apply only to LPP/LQL-based policies.
2. If you are using Terraform to automate policy management, you cannot use the Bulk Update Policy APIs to enable/disable policies.
3. Bulk operations are not compatible with config-analyzer (legacy) policies. They work only with LQL-based policies.

Enable/Disable Update Guidelines for Policies

In addition to the general guidelines, the following guidelines apply to the policies permitted in a single bulk update operation:

1. To update policy status (enabled/disabled), all policies in a single bulk operation must have the same policy type, either compliance or violation. That is, all policies in a single batch must be either compliance policies or all must be violation policies.
2. Manual policies cannot be bulk enabled/disabled.
3. For compliance or violation policies, bulk enable or disable works only if the policies all have the same combination of metadata tag values; that is, a policy with these tags can only be updated with other policies with the same tag values:

```
tags:
- security:compliance
- domain:AWS
```

4. For violation policies, bulk enabling or disabling policies is only allowed if the policy sources are the same, `CloudTrailRawEvents`, `Azure`, or `GCP`. That is, the source of violation policies in a batch cannot be different cloud provider types.

Severity Update Guidelines for Policies

The enable/disable update guidelines also apply to operations to update severity, but with these additional guidelines:

1. For compliance policies, all policies must also have the same original severity and metadata fields and tags, such as:

   ```
   Severity= medium
   ```

   And:

   ```
   tags:
   - security:compliance
   - domain:AWS
   - subdomain: configuration
   ```

2. For violation policies, all policies must have the same source and original severity value.

Bulk Update Example

The Bulk Update API enables you to change the severity or status (enabled/disabled) of multiple policies at a time. To use the Bulk Update API, send a list of policies with new field values to the Bulk API endpoint:

```
PATCH https://YourLacework.lacework.net/api/v2/Policies
```

A sample request body is as follows:

```
[
    {
        "policyId": "LW_Custom_UnrestrictedIngressToTCP445",
        "enabled": true,
        "severity": "high"
    },
    {
        "policyId": "LW_Custom_UnrestrictedIngressToTCP139",
        "enabled": true
    }
]
```

This example updates the `LW_Custom_UnrestrictedIngressToTCP445` policy with a new status and severity setting, along with a second policy, `LW_Custom_UnrestrictedIngressToTCP139`, with a new status value.

A successful response returns the full policy definition of each updated policy with new values. See Bulk Update Policies (beta) for more information about the API. Also note the following guidelines on the types of policies that can be submitted in a single batch.

Alert Profile Overview

An alert profile is a set of metadata that defines how your LQL queries are translated into alerts.

Alert profiles exist as a system. Lacework FortiCNAPP provides a set of predefined alert profiles to ensure that policy evaluation gives you useful results out of the box. To create your own customized profiles, extend an existing alert profile and add your custom templates to it.

An alert profile has three components:

- Fields
- Alert templates
- Description keys

The following sections discuss each component.

Fields

A *field* is a declaration of a field to be mapped in from an LQL query. Only LQL result fields that are declared as an alert profile field are mapped into event details and alerts. Fields returned by a query that are not listed as an alert profile field won't be mapped into event details and alerts.

For each Lacework FortiCNAPP-defined datasource, each field of that datasource is already defined as an alert profile field.

Currently, alert profile APIs do not support defining custom fields.

Alert Templates

An *alert template* is a definition of content to create from the results of a resource's policy violation. The event name, subject, and description contained in the alert appear in pushed alerts and in the Lacework FortiCNAPP Console.

An alert template's subject and description fields are not fixed text. You can specify customized subject and description fields by using regular text, which can refer to fields returned from the description keys within curly braces.

Each Lacework FortiCNAPP-defined datasource has defined default alerts. Your policies can use them without modification.

Alert profile APIs support creating and modifying your own custom alerts.

Description Keys

A *description key* is a placeholder variable that you can use in an alert template's subject and/or description. Description keys can refer to LQL query result fields and can also refer to other available data (such as metadata, like the name of a policy). Only description keys can be referred to by an alert template. A field must be used in a description key to be available in an alert.

For each Lacework FortiCNAPP-defined alert profile field, a description key is already defined for use in your alert templates.

Currently, alert profile APIs do not support defining custom description keys.

Defining Alert Templates in a Profile

To create new alert templates, create your own alert profile that extends a predefined Lacework FortiCNAPP alert profile.

Each alert has the following fields:

- `name`: The name that policies can use to refer to this template when generating alerts.
- `eventName`: The name of the resulting alert.
- `subject`: The subject text for the resulting alert.
- `description`: The description text for the resulting alert.

Each predefined alert profile contains default alert templates. If you do not define any custom alerts, a policy that references the alert profile will use the default alert template.

To use these alerts in a policy, refer to the alerts by the name you give them. The Lacework FortiCNAPP Policy Platform generates events and alerts based on the alert template that the policy refers to. If the policy refers to a named alert, but the alert profile doesn't have an alert with that name being generated, the policy uses the Lacework FortiCNAPP default template.

Select an Alert Profile

Because an alert profile helps to map what data is available for the alert message, each alert profile corresponds to a datasource. The following sections list the alert profiles and their corresponding datasources.

An alert profile has two components: the alert profile ID and the alert template name which follow this format: `alertProfileId.alert_template_name`.

For example, if you created a query that uses the `LW_CFG_AWS_EC2_SECURITY_GROUPS` datasource, use or extend the `LW_CFG_AWS_DEFAULT_PROFILE` alert profile.

To use the `LW_CFG_AWS_DEFAULT_PROFILE` alert profile, specify the following in the policies `alertProfile` field: `LW_CFG_AWS_DEFAULT_PROFILE.CFG_AWS_Violation`.

AWS Configuration Datasources

For all AWS configuration datasources, use the same alert profile.

Datasource	Alert Profile
`LW_CFG_AWS_*`	`LW_CFG_AWS_DEFAULT_PROFILE.CFG_AWS_Violation`

Agent Datasources

Datasource	Alert Profile
`LW_HA_DNS_REQUESTS`	`LW_HA_DNS_REQUESTS_DEFAULT_PROFILE.HA_DNS_Request_Violation`
`LW_HA_FILE_CHANGES`	`LW_HA_FILE_CHANGES_DEFAULT_PROFILE.HA_File_Changes_Violation`
`LW_HA_USER_LOGINS`	`LW_HA_USER_LOGINS_DEFAULT_PROFILE.HA_User_Login_Violation`
`LW_HE_CONTAINERS`	`LW_HE_CONTAINERS_DEFAULT_PROFILE.HE_Container_Violation`
`LW_HE_FILES`	`LW_HE_FILES_DEFAULT_PROFILE.HE_File_Violation`
`LW_HE_IMAGES`	`LW_HE_IMAGES_DEFAULT_PROFILE.HE_Image_Violation`
`LW_HE_MACHINES`	`LW_HE_MACHINES_DEFAULT_PROFILE.HE_Machine_Violation`
`LW_HE_PROCESSES`	`LW_HE_PROCESSES_DEFAULT_PROFILE.HE_Process_Violation`
`LW_HE_USERS`	`LW_HE_USERS_DEFAULT_PROFILE.HE_User_Violation`

AWS CloudTrail Datasource

Datasource	Alert Profile
`CloudTrailRawEvents`	`LW_CloudTrail_Alerts.CloudTrailDefaultAlert_AwsResource`

The API (`GET /api/v2/AlertProfiles`) does not currently expose the CloudTrail alert profile because it is not customizable.

Example Alert Profile

To get all alert profiles, use this endpoint:

```
GET https://AccountName.lacework.net/api/v2/AlertProfiles
```

To get only the `LW_CFG_AWS_DEFAULT_PROFILE` alert profile and its details, use this endpoint:

```
GET https://AccountName.lacework.net/api/v2/AlertProfiles/LW_CFG_AWS_DEFAULT_PROFILE
```

```
{
    "data": {
        "alertProfileId": "LW_CFG_AWS_DEFAULT_PROFILE",
        "extends": "LW_LPP_BaseProfile",
        "fields": [
            {
                "name": "_PRIMARY_TAG"
            },
            {
                "name": "RESOURCE_ID"
            }, ...
        ],
        "descriptionKeys": [
            {
                "name": "_OCCURRENCE",
                "spec": "{{_OCCURRENCE}}"
            },
            {
                "name": "RESOURCE_ID",
                "spec": "{{RESOURCE_ID}}"
            }, ...
        ],
        "alerts": [
            {
                "name": "CFG_AWS_PolicyChanged",
                "eventName": "LW Configuration AWS Violation Alert",
                "description": "{{_OCCURRENCE}} Violation for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}",
                "subject": "{{_OCCURRENCE}} violation detected for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}"
            },
            {
                "name": "CFG_AWS_NewViolation",
                "eventName": "LW Configuration AWS Violation Alert",
                "description": "{{_OCCURRENCE}} Violation for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}",
                "subject": "{{_OCCURRENCE}} violation detected for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}"
            },
            {
                "name": "CFG_AWS_Violation",
                "eventName": "LW Configuration AWS Violation Alert",
                "description": "{{_OCCURRENCE}} Violation for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}",
                "subject": "{{_OCCURRENCE}} violation detected for AWS Resource
{{RESOURCE_TYPE}}:{{RESOURCE_ID}} in account {{ACCOUNT_ID}} region {{RESOURCE_
REGION}}"
            }
        ]
```

```
          }
      }
```

Using the Resource Groups API

The Resource Groups API lets you view, modify, and create resource groups with conditions that allow you to closely control the resources that will be associated with the group. For example, you can create resource groups with resources with a particular tag or region.

This page provides examples of resource groups you can create with the Lacework API, and lists the fields that you can test with the conditions used to create them. For general information about resource groups, including how to work with them in the Lacework Console, see Resource Groups.

If you have defined Resource Groups in Terraform, see Convert Original To Newer Resource Groups in Terraform on page 39 for information about converting to the new format.

The Query Object

When creating or modifying a resource group, you can specify conditions for the resources that belong to that group using the `query` object in the request body to the endpoint.

The `query` object has this format:

```
"query":{
  "filters":[
    {
      "^\\w+$":{
        "field":"string",
        "operation":"STARTS_WITH",
        "values":[
           "string"
        ],
        "key":"string"
      }
    }
  ],
  "expression":{
    "operator":"AND",
    "children":[
      {
        "operator":"AND",
        "filterName":"string",
        "children":[ ]
      }
    ]
  }
}
```

The query object is made up of one or more filters, which specify a data field to be tested by the condition, an operation, and the values against which the field is tested based on the operation. The `key` field is required for fields that support key-value pairs, as described in Filterable Fields.

Possible operations include `STARTS_WITH`, `INCLUDES`, `ENDS_WITH`, and `EQUALS`. The resource groups field and the AWS account field work with `EQUALS` only.

The expression combines filters with an operator, AND or OR, into a multi-level logical hierarchy. Notice that there can only be one type of operator for each expression level. That is, the operator applied to filters at say the top expression level, or its child, or its child, can either be OR or AND, but not both.

Examples

The following example creates a resource group for all AWS resources in any US region that ends with 1:

```
{
  "name": "US regions 1",
  "description": "Resource in US regions ending with 1",
  "resourceType": "AWS",
  "query": {
    "filters": {
      "filter2": {
        "field": "Region",
        "operation": "STARTS_WITH",
        "values": [
          "us"
        ]
      },
      "filter3": {
        "field": "Region",
        "operation": "ENDS_WITH",
        "values": [
          "1"
        ]
      },
    },
    "expression": {
      "operator": "AND",
      "children": [
        {
          "filterName": "filter2"
        },
        {
          "filterName": "filter3"
        }
      ]
    }
  }
  "enabled": 1
}
```

The following example defines a group made up of resources associated with a specific account in either an Asia Pacific or US West (N. California) region:

```
{
  "NAME": "Resource group with nested conditions",
  "DESCRIPTION": "",
  "GROUP_TYPE": "AWS",
  "QUERY": {
    "filters": {
      "filter0": {
        "field": "Account",
        "operation": "EQUALS",
        "values": [
          "123412341234"
        ]
      },
      "filter77": {
        "field": "Region",
        "operation": "EQUALS",
        "values": [
          "ap-east-1"
        ]
      },
      "filter78": {
        "field": "Region",
        "operation": "EQUALS",
        "values": [
          "us-west-1"
        ]
      }
    },
    "expression": {
      "operator": "AND",
      "children": [
        {
          "filterName": "filter0"
        },
        {
          "operator": "OR",
          "children": [
            {
              "filterName": "filter77"
            },
            {
              "filterName": "filter78"
            }
          ]
        }
      ]
    }
  }
}
```

The following example creates a resource group for AWS resources with a HOST tag from the Asia Pacific South or Asia Pacific Southeast region:

```
{
  "name": "All resources tags and regions",
  "description": "Custom resource group for resources with a resource tag named HOST
with any value and from Asia Pacific South regions",
  "resourceType": "AWS",
  "query": {
    "filters": {
      "filter0": {
        "field": "Resource Tag",
        "operation": "INCLUDES",
        "values": [
          "*"
        ],
        "key": "HOST"
      },
      "filter1": {
        "field": "Region",
        "operation": "STARTS_WITH",
        "values": [
          "ap-south"
        ]
      }
    },
    "expression": {
      "operator": "AND",
      "children": [
        {
          "filterName": "filter0"
        },
        {
          "filterName": "filter1"
        }
      ]
    }
  },
  "enabled": 1
}
```

See the Resource Groups API for the complete specification of the request body and the response format.

Filterable Fields

The fields you can filter on in a `query` statement appear below, organized by resource type.

AWS

- Account
- Organization ID
- Resource Tag (requires a `key` field that specifies the resource tag name)
- Region
- Resource Group ID
- Resource Group Name

GCP

- Project ID
- Organization ID
- Organization Name
- Folder
- Resource Label (requires a `key` field that specifies the resource label name)
- Region
- Resource Group ID
- Resource Group Name

Azure

- Subscription ID
- Subscription Name
- Tenant ID
- Tenant Name
- Resource Tag (requires a `key` field that specifies the resource tag name)
- Region
- Resource Group ID
- Resource Group Name

Container

- Container Tag (requires a `key` field that specifies the container tag name)
- Container Label (requires a `key` field that specifies the container label name)
- Image Repo
- Image Registry
- Resource Group ID
- Resource Group Name

Machine

- Machine Tag (requires a `key` field that specifies the machine tag name)
- Resource Group ID
- Resource Group Name

Kubernetes

- AWS Account
- AWS Region

- GCP Project ID
- GCP Organization ID
- GCP Region
- CSP
- Cluster Name
- Namespace
- Resource Group ID
- Resource Group Name

Oracle Cloud Infrastructure (OCI)

- Compartment ID
- Compartment Name
- Region
- Resource Group ID
- Resource Group Name
- Resource Tag

Convert Original To Newer Resource Groups in Terraform

Resource groups provide a way to categorize assets, including the ability to populate resource groups based on conditional statements.

The original version of Resource Groups only supported account-based filters.

The new version of Resource Groups changes the Terraform syntax used to create resource groups. Where the original version of Resource Groups defined specific filter fields for each of the Resource Group types, the new version defines the `group` argument as an expression tree representing the relationships between resources using filters.

Refer to Filterable Fields in the Lacework FortiCNAPP API documentation for supported resource group filters.

- Lacework Account Resource Groups on page 40
- AWS Resource Groups on page 40
- Azure Resource Groups on page 40
- GCP Resource Groups on page 41
- Container Resource Groups on page 42
- Machine Resource Groups on page 43

The following examples illustrate some common Resource Group types. Additional Resource Group types may apply to your environment. These examples can be used to determine the converted format for those Resource Group types.

Lacework Account Resource Groups

Organization level resource groups are deprecated and not supported

AWS Resource Groups

Original Resource Groups

Previously, AWS Resource Groups only supported the `accounts` list field, which contained the accounts to be included in the resource group. If any of the accounts in the resource group matched an entry in the list, the resource group would be applied.

Example

```
resource "lacework_resource_group_aws" "example" {
  name        = "My AWS Resource Group"
  description = "A subset of AWS Accounts"
  accounts    = ["123456789", "234567891"]
}
```

New Resource Groups

The new Resource Groups support an expression tree structure in which the `field` field in a filter object is `Account` and the `value` field contains a list of the account IDs.

Example

```
resource "lacework_resource_group" "example" {
  name        = "My AWS Resource Group"
  type        = "AWS"
  description = "This groups a subset of AWS resources"
  group {
    operator = "OR"
    filter {
      filter_name = "filter1"
      field     = "Account"
      operation = "EQUALS"
      value     = ["123456789", "234567891"]
    }
  }
}
```

Azure Resource Groups

Original Resource Groups

Previously, Azure Resource Groups only supported the `tenant` field, which contained a list of subscriptions to be included in the Resource Group.

Example

```
resource "lacework_resource_group_azure" "example" {
  name          = "My Azure Resource Group"
  description   = "This groups a subset of Azure Subscriptions"
  tenant        = "a11aa1ab-111a-11ab-a000-11aa1111a11a"
  subscriptions = ["1a1a0b2-abc0-1ab1-1abc-1a000ab0a0a0", "2b000c3-ab10-1a01-1abc-
1a000ab0a0a0"]
}
```

New Resource Groups

The new Resource Groups support an expression tree structure in which the `field` field in a filter object is defined as either `Tenant ID` or `Subscription ID` and the `value` field contains a list of the tenant or subscrption IDs.

Example

```
resource "lacework_resource_group" "example" {
  name        = "My Azure Resource Group"
  type        = "AZURE"
  description = "This groups a subset of Azure Subscriptions"
  group {
    operator = "AND"
    filter {
      filter_name = "filter1"
      field       = "Tenant ID"
      operation = "EQUALS"
      value       = ["a11aa1ab-111a-11ab-a000-11aa1111a11a"]
    }
    group {
        operator = "OR"
        filter {
          filter_name = "filter2"
          field       = "Subscription ID"
          operation = "EQUALS"
          value       = ["1a1a0b2-abc0-1ab1-1abc-1a000ab0a0a0", "2b000c3-ab10-1a01-
1abc-1a000ab0a0a0"]
        }
    }
  }
}
```

GCP Resource Groups

Original Resource Groups

Previously, GCP Resource Groups only supported the `projects` field which contained a list of projects to be included in the resource group.

Example

```
resource "lacework_resource_group_gcp" "example" {
  name         = "My GCP Resource Group"
  description  = "This groups a subset of Gcp Projects"
  projects     = ["project-1", "project-2", "project-3"]
  organization = "MyGcpOrgID"
}
```

New Resource Groups

The new Resource Groups support an expression tree structure in which the `field` field in a filter object is defined as either `Organization ID` or `Project ID` and the `value` field contains a list of the organization or project IDs.

Example

```
resource "lacework_resource_group" "example" {
  name        = "My GCP Resource Group"
  type        = "GCP"
  description = "This groups a subset of Gcp Projects"
  group {
    operator = "AND"
    filter {
      filter_name = "filter1"
      field     = "Organization ID"
      operation = "EQUALS"
      value     = ["MyGcpOrgID"]
    }
    filter {
      filter_name = "filter2"
      field     = "Project ID"
      operation = "EQUALS"
      value     = ["project-1", "project-2", "project-3"]
    }
  }
}
```

Container Resource Groups

Original Resource Groups

Previously, Container Resource Groups supported the `container_tags` and `container_label` fields.

Example

```
resource "lacework_resource_group_container" "example" {
  name            = "My Container Resource Group"
  description     = "This groups a subset of Container Tags"
  container_tags = ["my-container"]
  container_label {
    key   = "name"
    value = "my-container"
  }
}
```

New Resource Groups

The new Resource Groups support an expression tree structure in which the `field` field in a filter object is defined as either `Image Tag` or `Container Label` and the `value` field contains a list of the image tags or container labels.

Example

```
resource "lacework_resource_group" "example" {
  name        = "My Container Resource Group"
  type        = "CONTAINER"
  description = "This groups a subset of Container Tags"
  group {
    operator = "AND"
    filter {
      filter_name = "filter1"
      field       = "Image Tag"
      operation   = "EQUALS"
      value       = ["my-container"]
   }
   filter {
      filter_name = "filter2"
      field       = "Container Label"
      operation   = "EQUALS"
      value       = ["my-container"]
      key         = "name"
   }
  }
}
```

Machine Resource Groups

Original Resource Groups

Previously, Machine Resource Groups supported the `machine_tags` field.

Example

```
resource "lacework_resource_group_machine" "example" {
  name        = "My Machine Resource Group"
  description = "This groups a subset of Machine Tags"
  machine_tags {
    key   = "name"
    value = "myMachine"
  }
}
```

New Resource Groups

The new Resource Groups support an expression tree structure in which the `field` field in a filter object is defined as `Machine Tag` and the `value` field contains a list of the machine tags.

Example

```
resource "lacework_resource_group" "example" {
  name        = "My Machine Resource Group"
  type        = "MACHINE"
  description = "This groups a subset of Machine Tags"
  group {
    operator = "AND"
    filter {
      filter_name = "filter1"
      field       = "Machine Tag"
      operation   = "EQUALS"
      value       = ["myMachine"]
      key         = "name"
    }
  }
}
```

Full Lacework FortiCNAPP API Reference

For the full browsable API reference, visit https://api.lacework.net/api/v2/docs.

www.fortinet.com