OpenShift Container Platform 4.18

Authorization APIs

Reference guide for authorization APIs

Last Updated: 2025-05-15

OpenShift Container Platform 4.18 Authorization APIs

Reference guide for authorization APIs

Legal Notice

Abstract

This document describes the OpenShift Container Platform authorization API objects and their detailed specifications.

Table of Contents

CHAPTER 1. AUTHORIZATION APIS

1.1. LOCALRESOURCEACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

LocalResourceAccessReview is a means to request a list of which users and groups are authorized to perform the action specified by spec in a particular namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.2. LOCALSUBJECTACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

LocalSubjectAccessReview is an object for requesting information about whether a user or group can perform an action in a particular namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.3. RESOURCEACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

ResourceAccessReview is a means to request a list of which users and groups are authorized to perform the action specified by spec
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.4. SELFSUBJECTRULESREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

SelfSubjectRulesReview is a resource you can create to determine which actions you can perform in a namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.5. SUBJECTACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

SubjectAccessReview is an object for requesting information about whether a user or group can perform an action
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.6. SUBJECTRULESREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

SubjectRulesReview is a resource you can create to determine which actions another user can perform in a namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.7. SELFSUBJECTREVIEW [AUTHENTICATION.K8S.IO/V1]

Description

SelfSubjectReview contains the user information that the kube-apiserver has about the user making this request. When using impersonation, users will receive the user info of the user being impersonated. If impersonation or request header authentication is used, any extra keys will have their case ignored and returned as lowercase.

Type

object

1.8. TOKENREQUEST [AUTHENTICATION.K8S.IO/V1]

Description

TokenRequest requests a token for a given service account.

Type

object

1.9. TOKENREVIEW [AUTHENTICATION.K8S.IO/V1]

Description

TokenReview attempts to authenticate a token to a known user. Note: TokenReview requests may be cached by the webhook token authenticator plugin in the kube-apiserver.

Type

object

1.10. LOCALSUBJECTACCESSREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

LocalSubjectAccessReview checks whether or not a user or group can perform an action in a given namespace. Having a namespace scoped resource makes it much easier to grant namespace scoped policy that includes permissions checking.

Type

object

1.11. SELFSUBJECTACCESSREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

SelfSubjectAccessReview checks whether or the current user can perform an action. Not filling in a spec.namespace means "in all namespaces". Self is a special case, because users should always be able to check whether they can perform an action

Type

object

1.12. SELFSUBJECTRULESREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

SelfSubjectRulesReview enumerates the set of actions the current user can perform within a namespace. The returned list of actions may be incomplete depending on the server's authorization mode, and any errors experienced during the evaluation. SelfSubjectRulesReview should be used by UIs to show/hide actions, or to quickly let an end user reason about their permissions. It should NOT Be used by external systems to drive authorization decisions as this raises confused deputy, cache lifetime/revocation, and correctness concerns. SubjectAccessReview, and LocalAccessReview are the correct way to defer authorization decisions to the API server.

Type

object

1.13. SUBJECTACCESSREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

SubjectAccessReview checks whether or not a user or group can perform an action.

Type

object

CHAPTER 2. LOCALRESOURCEACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

LocalResourceAccessReview is a means to request a list of which users and groups are authorized to perform the action specified by spec in a particular namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **namespace**
- **verb**
- **resourceAPIGroup**
- **resourceAPIVersion**
- **resource**
- **resourceName**
- **path**
- **isNonResourceURL**

2.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
content	**RawExtension**	Content is the actual content of the request for create and update
isNonResourceURL	**boolean**	IsNonResourceURL is true if this is a request for a non-resource URL (outside of the resource hierarchy)

Property	Type	Description
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces
path	**string**	Path is the path of a non resource URL
resource	**string**	Resource is one of the existing resource types
resourceAPIGroup	**string**	Group is the API group of the resource Serialized as resourceAPIGroup to avoid confusion with the 'groups' field when inlined
resourceAPIVersion	**string**	Version is the API version of the resource Serialized as resourceAPIVersion to avoid confusion with TypeMeta.apiVersion and ObjectMeta.resourceVersion when inlined
resourceName	**string**	ResourceName is the name of the resource being requested for a "get" or deleted for a "delete"
verb	**string**	Verb is one of: get, list, watch, create, update, delete

2.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.openshift.io/v1/namespaces/{namespace}/localresourceaccessreviews**
  - **POST**: create a LocalResourceAccessReview

2.2.1. /apis/authorization.openshift.io/v1/namespaces/{namespace}/localresourceacc

Table 2.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a LocalResourceAccessReview

Table 2.2. Body parameters

Parameter	Type	Description
body	**LocalResourceAccessReview** schema	

Table 2.3. HTTP responses

HTTP code	Reponse body
200 - OK	**LocalResourceAccessReview** schema
201 - Created	**LocalResourceAccessReview** schema
202 - Accepted	**LocalResourceAccessReview** schema
401 - Unauthorized	Empty

CHAPTER 3. LOCALSUBJECTACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

LocalSubjectAccessReview is an object for requesting information about whether a user or group can perform an action in a particular namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **namespace**
- **verb**
- **resourceAPIGroup**
- **resourceAPIVersion**
- **resource**
- **resourceName**
- **path**
- **isNonResourceURL**
- **user**
- **groups**
- **scopes**

3.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

Property	Type	Description
content	**RawExtension**	Content is the actual content of the request for create and update
groups	**array (string)**	Groups is optional. Groups is the list of groups to which the User belongs.
isNonResourceURL	**boolean**	IsNonResourceURL is true if this is a request for a non-resource URL (outside of the resource hierarchy)
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces
path	**string**	Path is the path of a non resource URL
resource	**string**	Resource is one of the existing resource types
resourceAPIGroup	**string**	Group is the API group of the resource Serialized as resourceAPIGroup to avoid confusion with the 'groups' field when inlined
resourceAPIVersion	**string**	Version is the API version of the resource Serialized as resourceAPIVersion to avoid confusion with TypeMeta.apiVersion and ObjectMeta.resourceVersion when inlined

Property	Type	Description
resourceName	**string**	ResourceName is the name of the resource being requested for a "get" or deleted for a "delete"
scopes	**array (string)**	Scopes to use for the evaluation. Empty means "use the unscoped (full) permissions of the user/groups". Nil for a self-SAR, means "use the scopes on this request". Nil for a regular SAR, means the same as empty.
user	**string**	User is optional. If both User and Groups are empty, the current authenticated user is used.
verb	**string**	Verb is one of: get, list, watch, create, update, delete

3.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.openshift.io/v1/namespaces/{namespace}/localsubjectaccessreviews**
  - **POST**: create a LocalSubjectAccessReview

3.2.1. /apis/authorization.openshift.io/v1/namespaces/{namespace}/localsubjectacce

Table 3.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a LocalSubjectAccessReview

Table 3.2. Body parameters

Parameter	Type	Description
body	**LocalSubjectAccessReview** schema	

Table 3.3. HTTP responses

HTTP code	Reponse body
200 - OK	**LocalSubjectAccessReview** schema
201 - Created	**LocalSubjectAccessReview** schema
202 - Accepted	**LocalSubjectAccessReview** schema
401 - Unauthorized	Empty

CHAPTER 4. RESOURCEACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

ResourceAccessReview is a means to request a list of which users and groups are authorized to perform the action specified by spec
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **namespace**
- **verb**
- **resourceAPIGroup**
- **resourceAPIVersion**
- **resource**
- **resourceName**
- **path**
- **isNonResourceURL**

4.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
content	**RawExtension**	Content is the actual content of the request for create and update
isNonResourceURL	**boolean**	IsNonResourceURL is true if this is a request for a non-resource URL (outside of the resource hierarchy)

Property	Type	Description
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces
path	**string**	Path is the path of a non resource URL
resource	**string**	Resource is one of the existing resource types
resourceAPIGroup	**string**	Group is the API group of the resource Serialized as resourceAPIGroup to avoid confusion with the 'groups' field when inlined
resourceAPIVersion	**string**	Version is the API version of the resource Serialized as resourceAPIVersion to avoid confusion with TypeMeta.apiVersion and ObjectMeta.resourceVersion when inlined
resourceName	**string**	ResourceName is the name of the resource being requested for a "get" or deleted for a "delete"
verb	**string**	Verb is one of: get, list, watch, create, update, delete

4.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.openshift.io/v1/resourceaccessreviews**
  - **POST**: create a ResourceAccessReview

4.2.1. /apis/authorization.openshift.io/v1/resourceaccessreviews

Table 4.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a ResourceAccessReview

Table 4.2. Body parameters

Parameter	Type	Description
body	**ResourceAccessReview** schema	

Table 4.3. HTTP responses

HTTP code	Reponse body
200 - OK	**ResourceAccessReview** schema
201 - Created	**ResourceAccessReview** schema
202 - Accepted	**ResourceAccessReview** schema
401 - Unauthorized	Empty

CHAPTER 5. SELFSUBJECTRULESREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

SelfSubjectRulesReview is a resource you can create to determine which actions you can perform in a namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

5.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
spec	**object**	SelfSubjectRulesReviewSpec adds information about how to conduct the check
status	**object**	SubjectRulesReviewStatus is contains the result of a rules check

5.1.1. .spec

Description

SelfSubjectRulesReviewSpec adds information about how to conduct the check

Type

object

Required

- **scopes**

Property	Type	Description
scopes	**array (string)**	Scopes to use for the evaluation. Empty means "use the unscoped (full) permissions of the user/groups". Nil means "use the scopes on this request".

5.1.2. .status

Description

SubjectRulesReviewStatus is contains the result of a rules check

Type

object

Required

- **rules**

Property	Type	Description
evaluationError	**string**	EvaluationError can appear in combination with Rules. It means some error happened during evaluation that may have prevented additional rules from being populated.
rules	**array**	Rules is the list of rules (no particular sort) that are allowed for the subject
rules[]	**object**	PolicyRule holds information that describes a policy rule, but does not contain information about who the rule applies to or which namespace the rule applies to.

5.1.3. .status.rules

Description

Rules is the list of rules (no particular sort) that are allowed for the subject

Type

array

5.1.4. .status.rules[]

Description

PolicyRule holds information that describes a policy rule, but does not contain information about who the rule applies to or which namespace the rule applies to.

Type

object

Required

- **verbs**
- **resources**

Property	Type	Description
apiGroups	**array (string)**	APIGroups is the name of the APIGroup that contains the resources. If this field is empty, then both kubernetes and origin API groups are assumed. That means that if an action is requested against one of the enumerated resources in either the kubernetes or the origin API group, the request will be allowed
attributeRestrictions	**RawExtension**	AttributeRestrictions will vary depending on what the Authorizer/AuthorizationAttribute Builder pair supports. If the Authorizer does not recognize how to handle the AttributeRestrictions, the Authorizer should report an error.

Property	Type	Description
nonResourceURLs	**array (string)**	NonResourceURLsSlice is a set of partial urls that a user should have access to. *s are allowed, but only as the full, final step in the path This name is intentionally different than the internal type so that the DefaultConvert works nicely and because the ordering may be different.
resourceNames	**array (string)**	ResourceNames is an optional white list of names that the rule applies to. An empty set means that everything is allowed.
resources	**array (string)**	Resources is a list of resources this rule applies to. ResourceAll represents all resources.
verbs	**array (string)**	Verbs is a list of Verbs that apply to ALL the ResourceKinds and AttributeRestrictions contained in this rule. VerbAll represents all kinds.

5.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.openshift.io/v1/namespaces/{namespace}/selfsubjectrulesreviews**
  - **POST**: create a SelfSubjectRulesReview

5.2.1. /apis/authorization.openshift.io/v1/namespaces/{namespace}/selfsubjectrulesr

Table 5.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SelfSubjectRulesReview

Table 5.2. Body parameters

Parameter	Type	Description
body	**SelfSubjectRulesReview** schema	

Table 5.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SelfSubjectRulesReview** schema
201 - Created	**SelfSubjectRulesReview** schema
202 - Accepted	**SelfSubjectRulesReview** schema
401 - Unauthorized	Empty

CHAPTER 6. SUBJECTACCESSREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

SubjectAccessReview is an object for requesting information about whether a user or group can perform an action
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **namespace**
- **verb**
- **resourceAPIGroup**
- **resourceAPIVersion**
- **resource**
- **resourceName**
- **path**
- **isNonResourceURL**
- **user**
- **groups**
- **scopes**

6.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

Property	Type	Description
content	**RawExtension**	Content is the actual content of the request for create and update
groups	**array (string)**	GroupsSlice is optional. Groups is the list of groups to which the User belongs.
isNonResourceURL	**boolean**	IsNonResourceURL is true if this is a request for a non-resource URL (outside of the resource hierarchy)
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces
path	**string**	Path is the path of a non resource URL
resource	**string**	Resource is one of the existing resource types
resourceAPIGroup	**string**	Group is the API group of the resource Serialized as resourceAPIGroup to avoid confusion with the 'groups' field when inlined
resourceAPIVersion	**string**	Version is the API version of the resource Serialized as resourceAPIVersion to avoid confusion with TypeMeta.apiVersion and ObjectMeta.resourceVersion when inlined

Property	Type	Description
resourceName	**string**	ResourceName is the name of the resource being requested for a "get" or deleted for a "delete"
scopes	**array (string)**	Scopes to use for the evaluation. Empty means "use the unscoped (full) permissions of the user/groups". Nil for a self-SAR, means "use the scopes on this request". Nil for a regular SAR, means the same as empty.
user	**string**	User is optional. If both User and Groups are empty, the current authenticated user is used.
verb	**string**	Verb is one of: get, list, watch, create, update, delete

6.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.openshift.io/v1/subjectaccessreviews**
  - **POST**: create a SubjectAccessReview

6.2.1. /apis/authorization.openshift.io/v1/subjectaccessreviews

Table 6.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SubjectAccessReview

Table 6.2. Body parameters

Parameter	Type	Description
body	**SubjectAccessReview** schema	

Table 6.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SubjectAccessReview** schema
201 - Created	**SubjectAccessReview** schema
202 - Accepted	**SubjectAccessReview** schema
401 - Unauthorized	Empty

CHAPTER 7. SUBJECTRULESREVIEW [AUTHORIZATION.OPENSHIFT.IO/V1]

Description

SubjectRulesReview is a resource you can create to determine which actions another user can perform in a namespace
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

7.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
spec	**object**	SubjectRulesReviewSpec adds information about how to conduct the check
status	**object**	SubjectRulesReviewStatus is contains the result of a rules check

7.1.1. .spec

Description

SubjectRulesReviewSpec adds information about how to conduct the check

Type

object

Required

- **user**
- **groups**
- **scopes**

Property	Type	Description
groups	**array (string)**	Groups is optional. Groups is the list of groups to which the User belongs. At least one of User and Groups must be specified.
scopes	**array (string)**	Scopes to use for the evaluation. Empty means "use the unscoped (full) permissions of the user/groups".
user	**string**	User is optional. At least one of User and Groups must be specified.

7.1.2. .status

Description

SubjectRulesReviewStatus is contains the result of a rules check

Type

object

Required

- **rules**

Property	Type	Description
evaluationError	**string**	EvaluationError can appear in combination with Rules. It means some error happened during evaluation that may have prevented additional rules from being populated.

Property	Type	Description
rules	**array**	Rules is the list of rules (no particular sort) that are allowed for the subject
rules[]	**object**	PolicyRule holds information that describes a policy rule, but does not contain information about who the rule applies to or which namespace the rule applies to.

7.1.3. .status.rules

Description

Rules is the list of rules (no particular sort) that are allowed for the subject

Type

array

7.1.4. .status.rules[]

Description

PolicyRule holds information that describes a policy rule, but does not contain information about who the rule applies to or which namespace the rule applies to.

Type

object

Required

- **verbs**
- **resources**

Property	Type	Description
apiGroups	**array (string)**	APIGroups is the name of the APIGroup that contains the resources. If this field is empty, then both kubernetes and origin API groups are assumed. That means that if an action is requested against one of the enumerated resources in either the kubernetes or the origin API group, the request will be allowed

Property	Type	Description
attributeRestrictions	**RawExtension**	AttributeRestrictions will vary depending on what the Authorizer/AuthorizationAttributeBuilder pair supports. If the Authorizer does not recognize how to handle the AttributeRestrictions, the Authorizer should report an error.
nonResourceURLs	**array (string)**	NonResourceURLsSlice is a set of partial urls that a user should have access to. *s are allowed, but only as the full, final step in the path This name is intentionally different than the internal type so that the DefaultConvert works nicely and because the ordering may be different.
resourceNames	**array (string)**	ResourceNames is an optional white list of names that the rule applies to. An empty set means that everything is allowed.
resources	**array (string)**	Resources is a list of resources this rule applies to. ResourceAll represents all resources.
verbs	**array (string)**	Verbs is a list of Verbs that apply to ALL the ResourceKinds and AttributeRestrictions contained in this rule. VerbAll represents all kinds.

7.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.openshift.io/v1/namespaces/{namespace}/subjectrulesreviews**
  - **POST**: create a SubjectRulesReview

7.2.1. /apis/authorization.openshift.io/v1/namespaces/{namespace}/subjectrulesrevie

Table 7.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SubjectRulesReview

Table 7.2. Body parameters

Parameter	Type	Description
body	**SubjectRulesReview** schema	

Table 7.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SubjectRulesReview** schema
201 - Created	**SubjectRulesReview** schema

HTTP code	Reponse body
202 - Accepted	**SubjectRulesReview** schema
401 - Unauthorized	Empty

CHAPTER 8. SELFSUBJECTREVIEW [AUTHENTICATION.K8S.IO/V1]

Description

SelfSubjectReview contains the user information that the kube-apiserver has about the user making this request. When using impersonation, users will receive the user info of the user being impersonated. If impersonation or request header authentication is used, any extra keys will have their case ignored and returned as lowercase.

Type

object

8.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
status	**object**	SelfSubjectReviewStatus is filled by the kube-apiserver and sent back to a user.

8.1.1. .status

Description

SelfSubjectReviewStatus is filled by the kube-apiserver and sent back to a user.

Type

object

Property	Type	Description
userInfo	**object**	UserInfo holds the information about the user needed to implement the user.Info interface.

8.1.2. .status.userInfo

Description

UserInfo holds the information about the user needed to implement the user.Info interface.

Type

object

Property	Type	Description
extra	**object**	Any additional information provided by the authenticator.
extra{}	**array (string)**	
groups	**array (string)**	The names of groups this user is a part of.
uid	**string**	A unique value that identifies this user across time. If this user is deleted and another user by the same name is added, they will have different UIDs.
username	**string**	The name that uniquely identifies this user among all active users.

8.1.3. .status.userInfo.extra

Description

Any additional information provided by the authenticator.

Type

object

8.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authentication.k8s.io/v1/selfsubjectreviews**
  - **POST**: create a SelfSubjectReview

8.2.1. /apis/authentication.k8s.io/v1/selfsubjectreviews

Table 8.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SelfSubjectReview

Table 8.2. Body parameters

Parameter	Type	Description
body	**SelfSubjectReview** schema	

Table 8.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SelfSubjectReview** schema
201 - Created	**SelfSubjectReview** schema
202 - Accepted	**SelfSubjectReview** schema
401 - Unauthorized	Empty

CHAPTER 9. TOKENREQUEST [AUTHENTICATION.K8S.IO/V1]

Description

TokenRequest requests a token for a given service account.

Type

object

Required

- **spec**

9.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	TokenRequestSpec contains client provided parameters of a token request.
status	**object**	TokenRequestStatus is the result of a token request.

9.1.1. .spec

Description

TokenRequestSpec contains client provided parameters of a token request.

Type

object

Required

- **audiences**

Property	Type	Description
audiences	**array (string)**	Audiences are the intendend audiences of the token. A recipient of a token must identify themself with an identifier in the list of audiences of the token, and otherwise should reject the token. A token issued for multiple audiences may be used to authenticate against any of the audiences listed but implies a high degree of trust between the target audiences.
boundObjectRef	**object**	BoundObjectReference is a reference to an object that a token is bound to.
expirationSeconds	**integer**	ExpirationSeconds is the requested duration of validity of the request. The token issuer may return a token with a different validity duration so a client needs to check the 'expiration' field in a response.

9.1.2. .spec.boundObjectRef

Description

BoundObjectReference is a reference to an object that a token is bound to.

Type

object

Property	Type	Description
apiVersion	**string**	API version of the referent.

Property	Type	Description
kind	**string**	Kind of the referent. Valid kinds are 'Pod' and 'Secret'.
name	**string**	Name of the referent.
uid	**string**	UID of the referent.

9.1.3. .status

Description

TokenRequestStatus is the result of a token request.

Type

object

Required

- **token**
- **expirationTimestamp**

Property	Type	Description
expirationTimestamp	Time	ExpirationTimestamp is the time of expiration of the returned token.
token	**string**	Token is the opaque bearer token.

9.2. API ENDPOINTS

The following API endpoints are available:

- **/api/v1/namespaces/{namespace}/serviceaccounts/{name}/token**
  - **POST**: create token of a ServiceAccount

9.2.1. /api/v1/namespaces/{namespace}/serviceaccounts/{name}/token

Table 9.1. Global path parameters

Parameter	Type	Description
name	**string**	name of the TokenRequest

Table 9.2. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create token of a ServiceAccount

Table 9.3. Body parameters

Parameter	Type	Description
body	**TokenRequest** schema	

Table 9.4. HTTP responses

HTTP code	Reponse body
200 - OK	**TokenRequest** schema
201 - Created	**TokenRequest** schema

HTTP code	Reponse body
202 - Accepted	**TokenRequest** schema
401 - Unauthorized	Empty

CHAPTER 10. TOKENREVIEW [AUTHENTICATION.K8S.IO/V1]

Description

TokenReview attempts to authenticate a token to a known user. Note: TokenReview requests may be cached by the webhook token authenticator plugin in the kube-apiserver.

Type

object

Required

- **spec**

10.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	TokenReviewSpec is a description of the token authentication request.

Property	Type	Description
status	**object**	TokenReviewStatus is the result of the token authentication request.

10.1.1. .spec

Description

TokenReviewSpec is a description of the token authentication request.

Type

object

Property	Type	Description
audiences	**array (string)**	Audiences is a list of the identifiers that the resource server presented with the token identifies as. Audience-aware token authenticators will verify that the token was intended for at least one of the audiences in this list. If no audiences are provided, the audience will default to the audience of the Kubernetes apiserver.
token	**string**	Token is the opaque bearer token.

10.1.2. .status

Description

TokenReviewStatus is the result of the token authentication request.

Type

object

Property	Type	Description

Property	Type	Description
audiences	**array (string)**	Audiences are audience identifiers chosen by the authenticator that are compatible with both the TokenReview and token. An identifier is any identifier in the intersection of the TokenReviewSpec audiences and the token's audiences. A client of the TokenReview API that sets the spec.audiences field should validate that a compatible audience identifier is returned in the status.audiences field to ensure that the TokenReview server is audience aware. If a TokenReview returns an empty status.audience field where status.authenticated is "true", the token is valid against the audience of the Kubernetes API server.
authenticated	**boolean**	Authenticated indicates that the token was associated with a known user.
error	**string**	Error indicates that the token couldn't be checked
user	**object**	UserInfo holds the information about the user needed to implement the user.Info interface.

10.1.3. .status.user

Description

UserInfo holds the information about the user needed to implement the user.Info interface.

Type

object

Property	Type	Description
extra	**object**	Any additional information provided by the authenticator.
extra{}	**array (string)**	

Property	Type	Description
groups	**array (string)**	The names of groups this user is a part of.
uid	**string**	A unique value that identifies this user across time. If this user is deleted and another user by the same name is added, they will have different UIDs.
username	**string**	The name that uniquely identifies this user among all active users.

10.1.4. .status.user.extra

Description

Any additional information provided by the authenticator.

Type

object

10.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/oauth.openshift.io/v1/tokenreviews**
  - **POST**: create a TokenReview
- **/apis/authentication.k8s.io/v1/tokenreviews**
  - **POST**: create a TokenReview

10.2.1. /apis/oauth.openshift.io/v1/tokenreviews

Table 10.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a TokenReview

Table 10.2. Body parameters

Parameter	Type	Description
body	**TokenReview** schema	

Table 10.3. HTTP responses

HTTP code	Reponse body
200 - OK	**TokenReview** schema
201 - Created	**TokenReview** schema
202 - Accepted	**TokenReview** schema
401 - Unauthorized	Empty

10.2.2. /apis/authentication.k8s.io/v1/tokenreviews

Table 10.4. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a TokenReview

Table 10.5. Body parameters

Parameter	Type	Description
body	**TokenReview** schema	

Table 10.6. HTTP responses

HTTP code	Reponse body
200 - OK	**TokenReview** schema
201 - Created	**TokenReview** schema
202 - Accepted	**TokenReview** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

CHAPTER 11. LOCALSUBJECTACCESSREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

LocalSubjectAccessReview checks whether or not a user or group can perform an action in a given namespace. Having a namespace scoped resource makes it much easier to grant namespace scoped policy that includes permissions checking.

Type

object

Required

- **spec**

11.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard list metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	SubjectAccessReviewSpec is a description of the access request. Exactly one of ResourceAuthorizationAttributes and NonResourceAuthorizationAttributes must be set
status	**object**	SubjectAccessReviewStatus

11.1.1. .spec

Description

SubjectAccessReviewSpec is a description of the access request. Exactly one of ResourceAuthorizationAttributes and NonResourceAuthorizationAttributes must be set

Type

object

Property	Type	Description
extra	**object**	Extra corresponds to the user.Info.GetExtra() method from the authenticator. Since that is input to the authorizer it needs a reflection here.
extra{}	**array (string)**	
groups	**array (string)**	Groups is the groups you're testing for.
nonResourceAttributes	**object**	NonResourceAttributes includes the authorization attributes available for non-resource requests to the Authorizer interface
resourceAttributes	**object**	ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface
uid	**string**	UID information about the requesting user.

Property	Type	Description
user	**string**	User is the user you're testing for. If you specify "User" but not "Groups", then is it interpreted as "What if User were not a member of any groups

11.1.2. .spec.extra

Description

Extra corresponds to the user.Info.GetExtra() method from the authenticator. Since that is input to the authorizer it needs a reflection here.

Type

object

11.1.3. .spec.nonResourceAttributes

Description

NonResourceAttributes includes the authorization attributes available for non-resource requests to the Authorizer interface

Type

object

Property	Type	Description
path	**string**	Path is the URL path of the request
verb	**string**	Verb is the standard HTTP verb

11.1.4. .spec.resourceAttributes

Description

ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

Type

object

Property	Type	Description

Property	Type	Description
fieldSelector	**object**	FieldSelectorAttributes indicates a field limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.
group	**string**	Group is the API Group of the Resource. "*" means all.

Property	Type	Description
labelSelector	**object**	LabelSelectorAttributes indicates a label limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.
name	**string**	Name is the name of the resource being requested for a "get" or deleted for a "delete". "" (empty) means all.
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces "" (empty) is defaulted for LocalSubjectAccessReviews "" (empty) is empty for cluster-scoped resources "" (empty) means "all" for namespace scoped resources from a SubjectAccessReview or SelfSubjectAccessReview

Property	Type	Description
resource	**string**	Resource is one of the existing resource types. "*" means all.
subresource	**string**	Subresource is one of the existing resource types. "" means none.
verb	**string**	Verb is a kubernetes resource API verb, like: get, list, watch, create, update, delete, proxy. "*" means all.
version	**string**	Version is the API Version of the Resource. "*" means all.

11.1.5. .spec.resourceAttributes.fieldSelector

Description

FieldSelectorAttributes indicates a field limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array (FieldSelectorRequirement)**	requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

11.1.6. .spec.resourceAttributes.labelSelector

Description

LabelSelectorAttributes indicates a label limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array (LabelSelectorRequirement)**	requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

11.1.7. .status

Description

SubjectAccessReviewStatus

Type

object

Required

- **allowed**

Property	Type	Description
allowed	**boolean**	Allowed is required. True if the action would be allowed, false otherwise.
denied	**boolean**	Denied is optional. True if the action would be denied, otherwise false. If both allowed is false and denied is false, then the authorizer has no opinion on whether to authorize the action. Denied may not be true if Allowed is true.
evaluationError	**string**	EvaluationError is an indication that some error occurred during the authorization check. It is entirely possible to get an error and be able to continue determine authorization status in spite of it. For instance, RBAC can be missing a role, but enough roles are still present and bound to reason about the request.

Property	Type	Description
reason	**string**	Reason is optional. It indicates why a request was allowed or denied.

11.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.k8s.io/v1/namespaces/{namespace}/localsubjectaccessreviews**
  - **POST**: create a LocalSubjectAccessReview

11.2.1. /apis/authorization.k8s.io/v1/namespaces/{namespace}/localsubjectaccessrevi

Table 11.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a LocalSubjectAccessReview

Table 11.2. Body parameters

Parameter	Type	Description
body	**LocalSubjectAccessReview** schema	

Table 11.3. HTTP responses

HTTP code	Reponse body
200 - OK	**LocalSubjectAccessReview** schema
201 - Created	**LocalSubjectAccessReview** schema
202 - Accepted	**LocalSubjectAccessReview** schema
401 - Unauthorized	Empty

CHAPTER 12. SELFSUBJECTACCESSREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

SelfSubjectAccessReview checks whether or the current user can perform an action. Not filling in a spec.namespace means "in all namespaces". Self is a special case, because users should always be able to check whether they can perform an action

Type

object

Required

- **spec**

12.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard list metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	SelfSubjectAccessReviewSpec is a description of the access request. Exactly one of ResourceAuthorizationAttributes and NonResourceAuthorizationAttributes must be set
status	**object**	SubjectAccessReviewStatus

12.1.1. .spec

Description

SelfSubjectAccessReviewSpec is a description of the access request. Exactly one of ResourceAuthorizationAttributes and NonResourceAuthorizationAttributes must be set

Type

object

Property	Type	Description
nonResourceAttributes	**object**	NonResourceAttributes includes the authorization attributes available for non-resource requests to the Authorizer interface
resourceAttributes	**object**	ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

12.1.2. .spec.nonResourceAttributes

Description

NonResourceAttributes includes the authorization attributes available for non-resource requests to the Authorizer interface

Type

object

Property	Type	Description
path	**string**	Path is the URL path of the request

Property	Type	Description
verb	**string**	Verb is the standard HTTP verb

12.1.3. .spec.resourceAttributes

Description

ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

Type

object

Property	Type	Description
fieldSelector	**object**	FieldSelectorAttributes indicates a field limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.
group	**string**	Group is the API Group of the Resource. "*" means all.

Property	Type	Description
labelSelector	**object**	LabelSelectorAttributes indicates a label limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.
name	**string**	Name is the name of the resource being requested for a "get" or deleted for a "delete". "" (empty) means all.
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces "" (empty) is defaulted for LocalSubjectAccessReviews "" (empty) is empty for cluster-scoped resources "" (empty) means "all" for namespace scoped resources from a SubjectAccessReview or SelfSubjectAccessReview

Property	Type	Description
resource	**string**	Resource is one of the existing resource types. "*" means all.
subresource	**string**	Subresource is one of the existing resource types. "" means none.
verb	**string**	Verb is a kubernetes resource API verb, like: get, list, watch, create, update, delete, proxy. "*" means all.
version	**string**	Version is the API Version of the Resource. "*" means all.

12.1.4. .spec.resourceAttributes.fieldSelector

Description

FieldSelectorAttributes indicates a field limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array (FieldSelectorRequirement)**	requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

12.1.5. .spec.resourceAttributes.labelSelector

Description

LabelSelectorAttributes indicates a label limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array (LabelSelectorRequirement)**	requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

12.1.6. .status

Description

SubjectAccessReviewStatus

Type

object

Required

- **allowed**

Property	Type	Description
allowed	**boolean**	Allowed is required. True if the action would be allowed, false otherwise.
denied	**boolean**	Denied is optional. True if the action would be denied, otherwise false. If both allowed is false and denied is false, then the authorizer has no opinion on whether to authorize the action. Denied may not be true if Allowed is true.
evaluationError	**string**	EvaluationError is an indication that some error occurred during the authorization check. It is entirely possible to get an error and be able to continue determine authorization status in spite of it. For instance, RBAC can be missing a role, but enough roles are still present and bound to reason about the request.

Property	Type	Description
reason	**string**	Reason is optional. It indicates why a request was allowed or denied.

12.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.k8s.io/v1/selfsubjectaccessreviews**
  - **POST**: create a SelfSubjectAccessReview

12.2.1. /apis/authorization.k8s.io/v1/selfsubjectaccessreviews

Table 12.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SelfSubjectAccessReview

Table 12.2. Body parameters

Parameter	Type	Description
body	**SelfSubjectAccessReview** schema	

Table 12.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SelfSubjectAccessReview** schema
201 - Created	**SelfSubjectAccessReview** schema
202 - Accepted	**SelfSubjectAccessReview** schema
401 - Unauthorized	Empty

CHAPTER 13. SELFSUBJECTRULESREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

SelfSubjectRulesReview enumerates the set of actions the current user can perform within a namespace. The returned list of actions may be incomplete depending on the server's authorization mode, and any errors experienced during the evaluation. SelfSubjectRulesReview should be used by UIs to show/hide actions, or to quickly let an end user reason about their permissions. It should NOT Be used by external systems to drive authorization decisions as this raises confused deputy, cache lifetime/revocation, and correctness concerns. SubjectAccessReview, and LocalAccessReview are the correct way to defer authorization decisions to the API server.

Type

object

Required

- **spec**

13.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard list metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	SelfSubjectRulesReviewSpec defines the specification for SelfSubjectRulesReview.
status	**object**	SubjectRulesReviewStatus contains the result of a rules check. This check can be incomplete depending on the set of authorizers the server is configured with and any errors experienced during evaluation. Because authorization rules are additive, if a rule appears in a list it's safe to assume the subject has that permission, even if that list is incomplete.

13.1.1. .spec

Description

SelfSubjectRulesReviewSpec defines the specification for SelfSubjectRulesReview.

Type

object

Property	Type	Description
namespace	**string**	Namespace to evaluate rules for. Required.

13.1.2. .status

Description

SubjectRulesReviewStatus contains the result of a rules check. This check can be incomplete depending on the set of authorizers the server is configured with and any errors experienced during evaluation. Because authorization rules are additive, if a rule appears in a list it's safe to assume the subject has that permission, even if that list is incomplete.

Type

object

Required

- **resourceRules**
- **nonResourceRules**
- **incomplete**

Property	Type	Description
evaluationError	**string**	EvaluationError can appear in combination with Rules. It indicates an error occurred during rule evaluation, such as an authorizer that doesn't support rule evaluation, and that ResourceRules and/or NonResourceRules may be incomplete.
incomplete	**boolean**	Incomplete is true when the rules returned by this call are incomplete. This is most commonly encountered when an authorizer, such as an external authorizer, doesn't support rules evaluation.
nonResourceRules	**array**	NonResourceRules is the list of actions the subject is allowed to perform on non-resources. The list ordering isn't significant, may contain duplicates, and possibly be incomplete.
nonResourceRules[]	**object**	NonResourceRule holds information that describes a rule for the non-resource
resourceRules	**array**	ResourceRules is the list of actions the subject is allowed to perform on resources. The list ordering isn't significant, may contain duplicates, and possibly be incomplete.
resourceRules[]	**object**	ResourceRule is the list of actions the subject is allowed to perform on resources. The list ordering isn't significant, may contain duplicates, and possibly be incomplete.

13.1.3. .status.nonResourceRules

Description

NonResourceRules is the list of actions the subject is allowed to perform on non-resources. The list ordering isn't significant, may contain duplicates, and possibly be incomplete.

Type

array

13.1.4. .status.nonResourceRules[]

Description

NonResourceRule holds information that describes a rule for the non-resource

Type

object

Required

- **verbs**

Property	Type	Description
nonResourceURLs	**array (string)**	NonResourceURLs is a set of partial urls that a user should have access to. **s are allowed, but only as the full, final step in the path.** "" means all.
verbs	**array (string)**	Verb is a list of kubernetes non-resource API verbs, like: get, post, put, delete, patch, head, options. "*" means all.

13.1.5. .status.resourceRules

Description

ResourceRules is the list of actions the subject is allowed to perform on resources. The list ordering isn't significant, may contain duplicates, and possibly be incomplete.

Type

array

13.1.6. .status.resourceRules[]

Description

ResourceRule is the list of actions the subject is allowed to perform on resources. The list ordering isn't significant, may contain duplicates, and possibly be incomplete.

Type

object

Required

- **verbs**

Property	Type	Description
apiGroups	**array (string)**	APIGroups is the name of the APIGroup that contains the resources. If multiple API groups are specified, any action requested against one of the enumerated resources in any API group will be allowed. "*" means all.
resourceNames	**array (string)**	ResourceNames is an optional white list of names that the rule applies to. An empty set means that everything is allowed. "*" means all.
resources	**array (string)**	Resources is a list of resources this rule applies to. "**" means all in the specified apiGroups. "**/foo" represents the subresource 'foo' for all resources in the specified apiGroups.
verbs	**array (string)**	Verb is a list of kubernetes resource API verbs, like: get, list, watch, create, update, delete, proxy. "*" means all.

13.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.k8s.io/v1/selfsubjectrulesreviews**
  - **POST**: create a SelfSubjectRulesReview

13.2.1. /apis/authorization.k8s.io/v1/selfsubjectrulesreviews

Table 13.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SelfSubjectRulesReview

Table 13.2. Body parameters

Parameter	Type	Description
body	**SelfSubjectRulesReview** schema	

Table 13.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SelfSubjectRulesReview** schema
201 - Created	**SelfSubjectRulesReview** schema
202 - Accepted	**SelfSubjectRulesReview** schema
401 - Unauthorized	Empty

CHAPTER 14. SUBJECTACCESSREVIEW [AUTHORIZATION.K8S.IO/V1]

Description

SubjectAccessReview checks whether or not a user or group can perform an action.

Type

object

Required

- **spec**

14.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard list metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	SubjectAccessReviewSpec is a description of the access request. Exactly one of ResourceAuthorizationAttributes and NonResourceAuthorizationAttributes must be set
status	**object**	SubjectAccessReviewStatus

14.1.1. .spec

Description

SubjectAccessReviewSpec is a description of the access request. Exactly one of ResourceAuthorizationAttributes and NonResourceAuthorizationAttributes must be set

Type

object

Property	Type	Description
extra	**object**	Extra corresponds to the user.Info.GetExtra() method from the authenticator. Since that is input to the authorizer it needs a reflection here.
extra{}	**array (string)**	
groups	**array (string)**	Groups is the groups you're testing for.
nonResourceAttributes	**object**	NonResourceAttributes includes the authorization attributes available for non-resource requests to the Authorizer interface
resourceAttributes	**object**	ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface
uid	**string**	UID information about the requesting user.

Property	Type	Description
user	**string**	User is the user you're testing for. If you specify "User" but not "Groups", then is it interpreted as "What if User were not a member of any groups

14.1.2. .spec.extra

Description

Extra corresponds to the user.Info.GetExtra() method from the authenticator. Since that is input to the authorizer it needs a reflection here.

Type

object

14.1.3. .spec.nonResourceAttributes

Description

NonResourceAttributes includes the authorization attributes available for non-resource requests to the Authorizer interface

Type

object

Property	Type	Description
path	**string**	Path is the URL path of the request
verb	**string**	Verb is the standard HTTP verb

14.1.4. .spec.resourceAttributes

Description

ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

Type

object

Property	Type	Description

Property	Type	Description
fieldSelector	**object**	FieldSelectorAttributes indicates a field limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.
group	**string**	Group is the API Group of the Resource. "*" means all.

Property	Type	Description
labelSelector	**object**	LabelSelectorAttributes indicates a label limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.
name	**string**	Name is the name of the resource being requested for a "get" or deleted for a "delete". "" (empty) means all.
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces "" (empty) is defaulted for LocalSubjectAccessReviews "" (empty) is empty for cluster-scoped resources "" (empty) means "all" for namespace scoped resources from a SubjectAccessReview or SelfSubjectAccessReview

Property	Type	Description
resource	**string**	Resource is one of the existing resource types. "*" means all.
subresource	**string**	Subresource is one of the existing resource types. "" means none.
verb	**string**	Verb is a kubernetes resource API verb, like: get, list, watch, create, update, delete, proxy. "*" means all.
version	**string**	Version is the API Version of the Resource. "*" means all.

14.1.5. .spec.resourceAttributes.fieldSelector

Description

FieldSelectorAttributes indicates a field limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array (FieldSelectorRequirement)**	requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

14.1.6. .spec.resourceAttributes.labelSelector

Description

LabelSelectorAttributes indicates a label limited access. Webhook authors are encouraged to * ensure rawSelector and requirements are not both set * consider the requirements field if set * not try to parse or consider the rawSelector field if set. This is to avoid another CVE-2022-2880 (i.e. getting different systems to agree on how exactly to parse a query is not something we want), see https://www.oxeye.io/resources/golang-parameter-smuggling-attack for more details. For the *SubjectAccessReview endpoints of the kube-apiserver: * If rawSelector is empty and requirements are empty, the request is not limited. * If rawSelector is present and requirements are empty, the rawSelector will be parsed and limited if the parsing succeeds. * If rawSelector is empty and requirements are present, the requirements should be honored * If rawSelector is present and requirements are present, the request is invalid.

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array (LabelSelectorRequirement)**	requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

14.1.7. .status

Description

SubjectAccessReviewStatus

Type

object

Required

- **allowed**

Property	Type	Description
allowed	**boolean**	Allowed is required. True if the action would be allowed, false otherwise.
denied	**boolean**	Denied is optional. True if the action would be denied, otherwise false. If both allowed is false and denied is false, then the authorizer has no opinion on whether to authorize the action. Denied may not be true if Allowed is true.
evaluationError	**string**	EvaluationError is an indication that some error occurred during the authorization check. It is entirely possible to get an error and be able to continue determine authorization status in spite of it. For instance, RBAC can be missing a role, but enough roles are still present and bound to reason about the request.

Property	Type	Description
reason	**string**	Reason is optional. It indicates why a request was allowed or denied.

14.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/authorization.k8s.io/v1/subjectaccessreviews**
  - **POST**: create a SubjectAccessReview

14.2.1. /apis/authorization.k8s.io/v1/subjectaccessreviews

Table 14.1. Global query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP method

POST

Description

create a SubjectAccessReview

Table 14.2. Body parameters

Parameter	Type	Description
body	**SubjectAccessReview** schema	

Table 14.3. HTTP responses

HTTP code	Reponse body
200 - OK	**SubjectAccessReview** schema
201 - Created	**SubjectAccessReview** schema
202 - Accepted	**SubjectAccessReview** schema
401 - Unauthorized	Empty