Integration Layer API ^{1.0.5}

Integration Layer consists of two service endpoints. DSL accepts data from applications and the integration layer performs synchronization of that data between eligible applications, indexing and collaboration with external CSPs, i.e. sends the data to external CSPs based on the data specific trust circles or the predefined [trust circle OR team] ID. DCL accepts data from external CSPs, authorizes that data based on data specific trust circles and the integration layer performs synchronization of that data between eligible applications.

Schemes

HTTPS

dsl ⌄

POST   **/v1/dsl/integrationData**   Process new integration data from application

---

Integration layer gets new data from an application. Synchronizes the data between eligible applications. Indexes the data into Elastic. Sends the data to external CSPs if toShare is set to true.

Parameters   Try it out

Name   Description

Name	Description

intDataObj * required

(body)

Example Value Model

```
{
  "dataParams": {
    "originCspId": "string",
    "originApplicationId": "string",
    "originRecordId": "string",
    "cspId": "string",
    "applicationId": "string",
    "recordId": "string",
    "dateTime": "string",
    "url": "string"
  },
  "sharingParams": {
    "toShare": true,
    "isExternal": true,
    "trustCircleId": [
      "string"
    ],
    "teamId": [
      "string"
    ]
  },
  "dataType": "event",
  "dataObject": "string"
}
```

Parameter content type

application/json

Responses

Response content type   application/json

Code	Description
200	*Successful operation*
400	*Malformed Integration Data Structure*

PUT   `/v1/dsl/integrationData`   Process integration data update from application

Integration layer gets updated data from an application. Synchronizes the data between eligible applications. Updates the existing indexed document in Elastic. Sends the data to external CSPs if toShare is set to true.

Parameters

<div style="text-align: right">

Try it out

</div>

Name	Description

intDataObj * required
 	
(body)
	Example Value Model
	
	<pre>{
	  "dataParams": {
	    "originCspId": "string",
	    "originApplicationId": "string",
	    "originRecordId": "string",
	    "cspId": "string",
	    "applicationId": "string",
	    "recordId": "string",
	    "dateTime": "string",
	    "url": "string"
	  },
	  "sharingParams": {
	    "toShare": true,
	    "isExternal": true,
	    "trustCircleId": [
	      "string"
	    ],
	    "teamId": [
	      "string"
	    ]
	  },
	  "dataType": "event",
	  "dataObject": "string"
	}</pre>
	
	Parameter content type
	
	application/json

Responses

Response content type   application/json

Code	Description
200	*Successful operation*
400	*Malformed Integration Data Structure*

DELETE   `/v1/dsl/integrationData`   Process integration data deletion in application

Integration layer gets deletion request from an application on a specific integration data record. Forwards the deletion request to eligible applications. toShare is always set to false. Deletes the existing indexed document from Elastic. No deletion requests are forwarded to external CSPs. dataObject is always null.

Parameters

Try it out

Name	Description

intDataObj * required
 	
(body)
	Example Value Model
	
	``json
	{
	  "dataParams": {
	    "originCspId": "string",
	    "originApplicationId": "string",
	    "originRecordId": "string",
	    "cspId": "string",
	    "applicationId": "string",
	    "recordId": "string",
	    "dateTime": "string",
	    "url": "string"
	  },
	  "sharingParams": {
	    "toShare": true,
	    "isExternal": true,
	    "trustCircleId": [
	      "string"
	    ],
	    "teamId": [
	      "string"
	    ]
	  },
	  "dataType": "event",
	  "dataObject": "string"
	}
	``
	
	Parameter content type
	
	application/json

Responses

Response content type   application/json

Code	Description
200	*Successful operation*

Code	Description
400	*Malformed Integration Data Structure*

dcl ⌄

POST   `/v1/dcl/integrationData`   Process new or updated integration data from external CSP

Integration layer gets new or updated data from external CSP. Authorizes the data based on data specific trust circles. Sets the isExternal flag to true. Synchronizes the data to principal application based on data type. Principal applications expected to emit updated data back to integration layer after debuplication (/dsl will handle emitions).

Parameters   [**Try it out**]

Name	Description

intDataObj * required
 	
(body)
	Example Value Model
	
	``json
	{
	  "dataParams": {
	    "originCspId": "string",
	    "originApplicationId": "string",
	    "originRecordId": "string",
	    "cspId": "string",
	    "applicationId": "string",
	    "recordId": "string",
	    "dateTime": "string",
	    "url": "string"
	  },
	  "sharingParams": {
	    "toShare": true,
	    "isExternal": true,
	    "trustCircleId": [
	      "string"
	    ],
	    "teamId": [
	      "string"
	    ]
	  },
	  "dataType": "event",
	  "dataObject": "string"
	}
	``
	
	Parameter content type
	
	[application/json]

Responses

Response content type application/json

Code	Description
200	*Successful operation*
400	*Malformed Integration Data Structure*
403	*CSP authorization failed. External CSP is not authorized to send specific data type*
415	*Unsupported data type. Local CSP does not support given data type. No application that handles given data type is installed*

Models ∨

IntegrationData {
    dataParams   {...}
    sharingParams   {...}
    dataType   {...}
    dataObject   **string($json)**
}

DataParams {
    description: origin[Csp/Application/Record]Id are only set by the application that initiates flow #1 (see SAD).
    In this case Origin[csp/application/record]Id and [csp/application/record]Id are equal.
    In the case where a principal application adapter gets data from external CSP and re-emitts them for indexing, [csp/application/record]Id are set to that of the principal application while Origin[csp/application/record]Id remain unchanged.

    originCspId*   **string**
    originApplicationId*   **string**
    originRecordId*   **string**
    cspId*   **string**
    applicationId*   **string**
    recordId*   **string**

```
        dateTime*               string($dateTime)
                                timestamp of the record emission

        url                     string($url)
                                reference to the emmitted record based on
                                [csp/application/record]Id.
                                (see SAD)

}
```

IntegrationDataType   string
Enum:

```
    [ event, threat, incident, vulnerability, artefact, chat, file, contact, trustCircle ]
```

SharingParams   {

```
    description:            only one of the trustCircleId or teamId should exist.
                            If none exists, sharing is done based on data specific
                            trust circles

    toShare*               boolean
    isExternal*            boolean
    trustCircleId
                              [
                           An array of strings with the trust circle uuids. For backwards
                           combatibility a single string value is also accepted. Empty
                           strings will result in validation error. If trustCircleId is not
                           null then teamId is disregarded

                           string]
    teamId
                              [
                           An array of string with the team uuids. For backwards
                           combatibility a single string value is also accepted. Empty
                           strings will result in validation error. If trustCircleId is not
                           null then teamId is disregarded

                           string]
}
```