IBM **Security** Discover and Classify

# ISDC & TITANIAM SPECTRA INTEGRATION GUIDE

**IBM SECURITY DISCOVER AND CLASSIFY** 

VERSION 3.5.0

# **TABLE OF CONTENTS**

Table of Contents	2
Overview	3
Application Flow	3
Prerequisites	4
App Installation, Upgrade & Removal	6
Installation	6
Upgrade	8
Removal	8
App Usage	9
Create Usecases using POST Usecases API call	9
Get the List of Existing Usecases Using API Call	10
Get the Stats of Existing Usecases Behavior for Sucessful Encryption Attemps	12
Get the Stats of Existing Usecases Behavior for Failed Encryption Attemps	13

# **OVERVIEW**

Titaniam Server is a Java application that can be configured as one of the following modules:

- Titaniam Encrypted Analytical Vault (further referred as Vault) is a NoSQL datastore. Store your most valuable data and conduct full-featured search analytics while keeping the data encrypted at all times.
- Titaniam Translation Service allows you to encrypt/ decrypt a variety of data (structured, unstructured, binaries). This product can be used to secure sensitive data or to translate data released from other Titaniam products.
- Titaniam Proxy for Object store (referred to as Proxy in this document) allows you to protect an object store.
- Titaniam Elasticsearch Proxy allows you to protect an Elasticsearch node.

The goal of ISDC integration with Titaniam Spectra is to automate encryption of the sensitive files in Amazon S3 buckets.

USE CASE	DESCRIPTION
Automatic encryption of files containing personal data hosted in Amazon S3 bucket	Some S3 buckets may have many files containing plain text sensitive information. After ISDC analyzes the S3 bucket and discovers personal data in files stored there, the integration app will automatically encrypt these files.
File tagging/labeling for encrypted objects in S3 buckets	To provide context for the encrypted files in the S3 bucket, the integration app can assign tags (labels) to the files with useful information. For example, data elements and categories discovered by ISDC in the file.
Reports on workflow behavior, metrics of encrypted files, amount of pass/fail and reasons for failed results	The integration app can keep track on progress of the workflow (how many files Titaniam tried to encrypt and how successful it was), as well as log the affected resources.

Table 1: Use cases for ISDC integration with Titaniam Spectra

#### APPLICATION FLOW

The integration app is automated. A user manually configures a usecase according to their specific requirements. Then the usecase triggers the following automatic flow:

1. ISDC analyzes the Amazon S3 bucket and discovers personal data in files. These files must be encrypted.

2. Integration app receives the personal data locations (sources) hosted in S3 buckets and processes them according to the usecase scenario.

3. All sources related to the usecase are fetched into the Titaniam tool to manage encryption of the files in S3 bucket.

4. Once the files are encrypted, the integration app receives the status of the encryption request per source.

5. The integration app tags the S3 object: it assigns the data elements and categories found in the object as tags.



**Usecase** is a rule defined for integration package via API that defines the conditions of the workflow. For example, a user can configure usecases to encrypt only the files with plain text Credit Card Numbers or to encrypt all files with any senstive information in a specific bucket. These usecases can be dynamically enabled and disabled.

A usecase has a set of mandatory and optional configurations that define the workflow:

- Data Elements are a list of ISDC data element names that trigger the usecase execution. The data element names are dynamically updated according to ISDC settings. The integration app supports the out-of-box and custom data elements.
- Source URL is a string that filters the discovery results for presence of specific bucket to work with.



Figure 2: Flow of ISDC integration with Titaniam Spectra

#### PREREQUISITES

1. Titaniam Spectra installed:

- For Docker deployment, follow the <u>Production Installation Using Docker on EC2 Instance</u>.
- For K8s deployment, follow the <u>Production Installation Using Kubernetes</u>.

2. ISDC v3.4+ installed and having access to Titaniam Spectra instance:

#### telnet <titaniam\_host> <titaniam\_port>

3. AWS access key ID and secrets generated with the following permissions for the targetted S3 buckets (or all of them depending on usecase):

- s3:ListBucket
- s3:\*Object
- s3:GetBucketVersioning

- s3:GetObjectTagging
- s3:PutObjectTagging

4. Integration application installed. Follow the installation steps.

# **APP INSTALLATION, UPGRADE & REMOVAL**

#### **INSTALLATION**

Integration package is provided on demand. It can be installed on your VM, local machine or cluster node. To deploy the project, perform the following steps:

1. Extract the package directory using one of commands below depending on the archive extension (zip, tar, tar.gz):

#### unzip titaniam-integration.zip

or

#### tar -xvf titaniam-integration.tar

or

#### tar -xvf titaniam-integration.tar.gz

2. Change directory to extracted folder:

#### cd titaniam-integration-master

3. Copy the ./env.compose.example file, rename it to env.compose:

#### cp .env.compose.example .env.compose

4. Fill the file with data. See the sample configuration below:

Sample .env.compose file:

```
APP_NAME=titaniam-integration-1
APP_HOST=10.192.192.117
JWK_JWT=secret
```

KAFKA\_HOST=10.192.192.131 KAFKA\_PORT=30828 KAFKA\_USERNAME=kadmin KAFKA\_PASSWORD=kafkapassword SECURITY\_PROTOCOL=SASL\_SSL

TITANIUM\_HOST=18.206.252.64 TITANIUM\_PORT=8080

#### TITANIUM\_SCHEME=http

AWS\_ACCESS\_KEY\_ID=AKIARMCLMCOXUUXAT7ED AWS\_SECRET\_ACCESS\_KEY-Y=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

POSTGRES\_HOST=integration\_postgres\_splunk POSTGRES\_PORT=5432 POSTGRES\_DB=integration\_db POSTGRES\_USER=postgres POSTGRES\_PASSWORD=postgres POSTGRES\_POOL\_MAX\_CONN=3

5. You need to change APP\_HOST, KAFKA\_HOST, KAFKA\_PORT and titaniam details (host, port scheme and AWS access key).

5.1. To learn the host and port of Kafka, run this command:

#### kubectl get svc -n {YOUR\_NAMESPACE} kafka-cm-tls-0-external

Value under EXTERNAL IP is the KAFKA\_HOST. The KAFKA\_PORT is the value next to port 9094: 9094:<value>.

5.2. KAFKA\_HOST is the IP of appliance, which you can find as follows:

#### kubectl get nodes -n {YOUR\_NAMESPACE}

In the output, use the node IP address.

5.3. App\_host is hostname or IP address of ISDC you wich to import data from.

6. Start the stack:

#### docker-compose up -d --build

7. After the stack is started you should be able to reach the API swagger of the integration on https://<APP\_HOST>:5005/docs.

GET	/settings/titanium Get Nanium application settings	~
	······································	
Return ap	plication settings in JSON format	
Paramete	YS	Try it out
No porem		
NO param		
Response	85	
Code	Description	Links
200	Sussessful Despases	No links
	Media type application/json	
	Controls Accept header.	
	Example Value   Schema	
	{ "nome": "string", "http://www.string",	
	"description": "string", "logoDark": "string", "logoLight": "string".	
	"configurations": { "enableTestConnection": true,	
	"inputs": [ { "fd": "rtenina"	
	"lobel": "string", "ploceholde": "string",	
	"type": "string", "description": "string",	
	"required": true, "defaultYalue": "string", "#sytConfigurations". J	
	"multiselectConfigurations": {}, "selectConfigurations": {}	
404	}	No Vieles
401	Unauthorized	NO IINKS
	Media type	

Figure 1: API swagger of the integration

#### UPGRADE

To install newer version of application, the process is identical to installation. The docker volumes removal may be required, if explicitly specified.

#### REMOVAL

1. To delete the application, stop the containers from the directory with deployment files:

cd titaniam-integration-master docker-compose down	
---	--

2. Remove the docker volumes created for the project. Volumes can be listed using command:

docker volume rm \${docker volume ls -q |grep titaniam}

### **APP USAGE**

#### CREATE USECASES USING POST USECASES API CALL

#### Sample request

```
curl -X 'POST' \
 'https://<APP HOST>:5005/usecases' \
 -H 'accept: application/json' \
 -H 'Content-Type: application/json' \
 -d '{
 "name": "Test Usecase 1",
 "description": "Describe what this use case should cover",
 "data elements": [
  'CC_NUMBER',
  'ACCOUNT_ID',
  'PHONE NUMBER'
 ],
 "source url": "s3://",
 "action": "encrypt",
 "active": true
}'
```

Sample response

Success

```
{
"success": true
}
```

Fail (for example, incorrect data elements input syntax)

```
{
    "detail": [
    {
        "loc": [
        "body",
        116
     ],
        "msg": "Expecting value: line 5 column 5 (char 116)",
        "type": "value_error.jsondecode",
        "ctx": {
        "msg": "Expecting value",
        "doc": "{\n \"name\": \"Test Usecase 1\",\n \"description\": \"Describe what this use
        case should cover\",\n \"data_elements\": [\n 'CC_NUMBER',\n 'ACCOUNT_
        ID',\n 'PHONE_NUMBER'\n ],\n \"source_url\": \"s3://",\n \"action\":
        \"encrypt\",\n \"active\": true\n}",
    }
```

"pos": 116, "lineno": 5, "colno": 5 } } }	
POST /usecases Add application new filter settings	^
Parameters	Cancel Reset
No parameters	
Request body required	application/json ×
<pre>{     "name": "Test Usacsse 1",     "description": "Describe what this use case should cover",     "dist_cleanst": (     "CC_NUMBER",     "CC_NUMBER",     "PHONE_NUMBER" },     "source_url": "s3://",     "active": true }</pre>	*
Execute	Clear
Responses	
Curl Curl - X 'POST' \ 'https://10.192.192.117:5005/usecases' \ + 4 'Content-Type: application/json' \ - 4 ' { "dota_clements": [ "dota_clements": [ "dota_clements": [ "dota_clements": [ "CONMERT"] - *COUNT_IO", "ACCOUNT	
{ success": true }	Download

Figure 1: Example of usecase creation

### GET THE LIST OF EXISTING USECASES USING API CALL

#### Sample request

```
curl -X 'GET' \
'https://10.192.192.117:5005/usecases/titanium' \
-H 'accept: application/json'
```

#### Sample Response

```
[
    {
        "id": 67,
        "name": "Titaniam Use case",
        "description": "string",
        "action": "encrypt",
        "active": true,
        "counter": 24307,
        "success_counter": 6,
        "failed_counter": 24297
    }
]
```

	Table 2: Response parameters for existing usecases request			
PARAMETER	DESCRIPTION			
id	Unique identifier of usecase entity.			
name	Usecase alias.			
description	Usecase description.			
action	Type of action on source location. For Titaniam, only <i>encrypt</i> is available.			
active	Indicates whether the usecase is processing data. Options: true; false.			
counter	Metric for the number of source records processed in total. Numeric.			
success_counter	Number of files that were successfully encrypted.			
failed_counter	Number of files that the system failed to encrypt due to various reasons.			
GET /usecases/{usecas	GET /usecases/{usecase_type} Get application filters settings			

Return application filters settings in JSON format	
Parameters	Cancel
Name Description usecase_type * required titanium ~ string (path)	
Execute	Clear
Responses	
Curl curl -X 'GET' \ 'https://10.192.192.117:5005/usecases/titanium' \ -H 'accept: application/json' Request URL https://10.192.192.117:5005/usecases/titanium	₿.
Server response	
Code Details	
<pre>200 Response body [ {</pre>	Download

Figure 3: Example of usecase list request

# GET THE STATS OF EXISTING USECASES BEHAVIOR FOR SUCESSFUL ENCRYPTION ATTEMPS

#### Sample Request

curl -X 'GET' \
 'https://10.192.192.117:5005/get\_encrypted\_files?limit=1000&offset=0&encrypted=true' \
 -H 'accept: application/json'

Sample Response

Table 4: Response parameters for sucessful encryption attemps

PARAMETER	DESCRIPTION
encrypted	Current state of file encryption request. Options: true; false.
error_description	Applicable if the encryption attempt failed.
id	Unique identifier of the resource to encrypt.
path	Full path to resource.

GET	/get_encrypted_files Get encrypted files data from database	^			
Get inventa Query parar • limit: ; • offset • encry	a data from database ims: mumber of records, default = 1000 t: number of skipped limits, default = 0 prod: boolean vaniable, default = false				
Parameters	s	Cancel			
Name	Description				
limit integer (query)	1000				
offset	0				
(query) encrypted boolean (query)	true ~				
	Execute	Clear			
Baspansos					
nesponses	9 				
Curl curl -X '( 'https:/ -H 'acci	GET'\ //10.192.117:5005/get_encrypted_files?limit=1000&offset=0&encrypted=true'\ ept: application/json'	ß			
Request URL	L 10.192.192.117:5005/mst_encrypted_files?limit=1000&offset=D&encrypted=true				
Server respo	Server response				
Code	Details				
200	Response body				

Figure 5: Example of stats for successful encryption attemps

# GET THE STATS OF EXISTING USECASES BEHAVIOR FOR FAILED ENCRYPTION ATTEMPS

#### Sample Request

```
curl -X 'GET' \
    'https://10.192.192.117:5005/get_encrypted_files?lim-
it=1000&offset=0&encrypted=false' \
    -H 'accept: application/json'
```

#### Sample Response



```
"encrypted": false,
  "error description": "The bucket you are attempting to access must be addressed
using the specified endpoint. Please send all future requests to this endpoint. (Service:
S3, Status Code: 301, Request ID: WG4XXJ2GYVV6Q6Y3, Extended Request ID:
oQ5bVaONJ6S+FvAOaNtSVE/KMt3ip4cMEHUig17iigEEh0SqJQ76/s0HACxqkOVUBQ-
hwhhmYU28=)",
  "id": 1,
  "path": "1touch-athena-results/0e52ae62-f9ca-4564-9c69-aace234dc81d.csv"
},
  "encrypted": false,
  "error description": "The bucket you are attempting to access must be addressed
using the specified endpoint. Please send all future requests to this endpoint. (Service:
S3, Status Code: 301, Request ID: T6CXKJN2XY33196R, Extended Request ID:
ICeOJSmuADe76Gma-
gtxDdZV73K6L36HEOkooS3OpsIOYgeHLqyVBf93/mONuWrp+GJ16pbA2r0Q=)",
  "id": 2.
  "path": "1touch-athena-results/0f645974-d011-41b2-8f0c-edf947b8e57a.csv"
},
  "encrypted": false,
  "error description": "The bucket you are attempting to access must be addressed
using the specified endpoint. Please send all future requests to this endpoint. (Service:
S3, Status Code: 301, Request ID: 3CRC9BWH73GK4V67, Extended Request ID:
oU6ed8xUtGwNDqM2Yx2uCAY3Z4L4BtTVb2CdyUo1xK468d3JLjMTEk74NU4iEQv4X-
Dcy3vGdlc0=)",
  "id": 3.
  "path": "1touch-athena-results/09d1a304-8650-494f-9850-2956b2d410d5.csv"
},
 . . .
 ...
 . . .
```

PARAMETER	DESCRIPTION
encrypted	Current state of file encryption request. Options: true; false.
error_description	Reason for failed encryption attempt.
id	Unique identifier of the resource to encrypt.
path	Full path to resource.

Table 6: Response parameters for failed encryption attemps

GET	/get_encrypted_files Get encrypted files data from database				
Got invorto	n data faan datahasa				
Query para	a data mini database				
<ul> <li>limit:</li> </ul>	under of records, default = 1000				
<ul> <li>offse</li> <li>encry</li> </ul>	t: number of skipped limits, default = 0 ppted: boolean variable, default = false				
Parameter	Cancel				
Name	Description				
limit	1000				
integer (query)					
offset					
integer	0				
(query)					
boolean	faise ~				
(query)					
	Execute Clear				
Response	S				
Curl					
curl -X '	'GET' \				
'https: -H 'acc	://10.192.192.117:5005/get_encrypted_files?limit=1000&offset=0&encrypted=false' \ cept: application/json'				
Request UR	L.				
https://	10.192.192.117:5005/get_encrypted_files?limit=1000&offset=0&encrypted=false				
Server respo	onse				
Code	Details				
200	Response body				
	t "encrypted": false, "error description": "The hucket you are attempting to access must be addressed using the specified endpoint. Please send all future requests to this endpoint. (Service: Sa				
	ervor_description : ine bucket you are attempting to access must be adaressed using the specified endpoint. Piedse send all future requests to this endpoint. (Service: S3, Status Code: 301, Request ID: WG4XXJ2GYVKQ6Y3, Extended Request ID: oQ5bVaONJ6S+FvADaNtSVE/KMt3ip4cMEHUig17iiqEEh0SqJQ76/s0HACxqKOVUBQhwhhmYU28=)", "id": 1.				
	"path": "ltouch-athena-results/0e52ae62-f9ca-4564-9c69-aace234dc81d.csv" },				
	{ "encrypted": false,				
	"error_description": "The bucket you are attempting to access must be addressed using the specified endpoint. Please send all future requests to this endpoint. (Service: S3, Status Code: 301, Request ID: T6CKKJN2XY33196R, Extended Request ID: 1CeOJSmuADe76GmggtxD4ZV73K6L3GHEOkooS30psIOYqeHLqvVBf93/mDNuWhr+GJ16pbA2r0Q+)",				
	"id": 2, "path": "ltouch-athena-results/0f645974-d011-41b2-8f0c-edf947b8e57a.csv"				

#### Figure 7: Example of stats for failed encryption attemps

Result of the operations above is an encrypted file on S3 bucket and assigned tags specifiying the context around encryption:

https://dev-titaniam-integration.s3.eu-west-1.amazonaws.com/jjsmith.csv			🖺 Save 🗸	1	۶ 📮	
GET	GET v https://dev-titaniam-integration.s3.eu-west-1.amazonaws.com/jjsmith.csv			Sen	d ~	
Params	Authorization  Headers (9) Body Pre-request Script Te	sts Settings			Cookies	
Query	Params					
	Key	Value	Description	000	Bulk Edit	
	Кеу	Value	Description			
Body Prett	Cookies Headers (11) Test Results y Raw Preview Visualize Text ~ ====		B Status: 200 OK Time: 688 ms Size: 219.82 KB 1	Save Res	ponse ~ Im Q	
2 3 4 5 6 7 8 9 10 11 12 12 13 14 15 16 17 18	<pre>p 666636 669,66163_2)%0660406 6. Vo: 56016 0.&gt; 667 4.6 0 66- 7006 6. Vo: 56016 0.&gt; 667 4.6 0 66- 7006 6. Vo: 56016 0.&gt; 667 4.6 0 66- 7006 70060060606 (66 96600 NOSpe60) [36065560 6 0 5-0 6609,6700 205070 )44 Vj6 N6660 6 0 666560 6 170665166(3 6.6 0 0 0 0 0 6660 - 6 6666 6 6666) [6 6 6 0 0 6 556 6 +++; 5667c512 6 666 ,6676] V0(61830,V0(5)26 5 6 ++; 5667c512 6 666 ,6676) V0(61830,V0(5)26 5 0 + 5660 + 55660 + 55660 + 56660 + 56660 + 56660 + 56660 + 56660 + 56660 + 56660 + 56660 + 56600 + 5660 + 566000 + 56600 + 56600 + 56600 + 566000 + 566000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 + 56000 +</pre>	8d <t #8666666666666="" 0±805665-160="" 61'<br="" 6607"l6028="" 86="" tik9="" ±e6660num48="">8L5283361 655126608*366,57TNy535F 815,97T + 8573 E0 R k66*66 8666298 660666666666666308 ± f.0 *st6656 05CD2;8d3Ak36h6ey*6AD 8 66666666×366 056626660 ± 6 *s10 6664 66/66660 10 66           0 606666668566668         0 *s166566         0 *s166566         0 *s1665666         0 *s16656666           0 606666668566         0 *s166566         0 *s16656666         0 *s16656666         0 *s16656666         0 *s1665666         0 *s1665666         0 *s16656666         0 *s166566666         0 *s1665667666         0 *s16656676666         0 *s166567666666666666666666666666666666666</t>	200 v-000 04000,000 00000070002X-00100%000 [D#DDD10000270)0 0,02m+5)50-005000 000000000000000 000000000000	<b>ið</b> ) &		

#### Figure 8: Encrypted file output

Tags (3) Trick storage cost of other enteria by tagging year abjects. Learn more 🖉		Edit
Key	Value	
key-signature	dev-titaniam-integration	
inventa-categories	government-issued-ids political vehicle-related-information professional-information demographic religion financial-information race digital-identification personal-information contact-information	
inventa-entity- types	HIDDLE JUITLAL CC_EXPRES ARE CC_JUNRIER CITY CC_TYPE PHONE_JUMBER SURIAME COLUTRY GYEN, JAME GUID DRIVER_LICENE USERNAME FULL JAME BMAIL_ADDRESS VEHICLE_YEAR BIRTHOAY COLUTRY_CODE POLITICAL_OPNION COMPANY DOMAIN TITLE CC_CVV	

Figure 9: S3 resource tagging

IBM, the IBM logo, and IBM Security Discover and Classify are trademarks or registered trademarks of International Business Machines Corporation, in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on ibm.com/trademark.