z/OS MVS IPCS Customization (SA23-1383-00)

Topic: IPCS Installation Package

SubTopic: Customizing Data Privacy for Diagnostics

Customizing Data Privacy for Diagnostics

The Data Privacy for Diagnostics Analyzer provides the facilities to scan and identify data within dumps that may be sensitive personal information (SPI). The Data Privacy for Diagnostics Analyzer runs via batch jobs and utilizes the zFS file system to retain all its required input, and as a repository for its reports. Required inputs include dictionaries used to identify SPI. Reports can be generated to help a user understand what caused pages to be flagged as containing sensitive data. Users will be able to provide feedback by updating this information in the file system and running the feedback analysis tool to improve the SPI data detection/analysis. SPI data detection is achieved through the use of grammar tokens. An input token is a set of printable characters that are separated by a delimiter of a space or non-printable character. Tokens are then matched against built-in or custom identifiers to determine if they are deemed as sensitive. Customization of the Data Privacy for Diagnostics Analyzer can tailor which tokens are to be flagged as sensitive data or treated as non-sensitive information.

In order to use the Analyzer, some initial set up must be performed. To help with this set up, a sample batch job has been provided. The batch job will create and initialize the file system, mount it to the desired mount point, and run an initialization shell script. See 'SYS1.SAMPLIB(BLSDPJIN)' for instructions on how to modify the sample batch job example to run it on a different system(s).

One consideration that should be given is to the access control for this file system. Some of the sub-directories may contain sensitive data that has been extracted from dumps, or data that has been ingested by your customization for dump analysis. This data may be in reports, in files after feedback has been given and data has been ingested. Therefore, you want to ensure that only intended personnel have access to these folders. The sample JCL BLSDPJIN assigns the permission to the directory with the MODE parameter on the MKDIR statement that defines the file system directory, which defaults to (7,5,0). One option for you is to use FSACCESS to control user access to this data. See **Steps for restricting access to a zFS file system** in "**z/OS Security Server RACF Security Administrator's Guide**" for more information on using FSACCESS to control access to file systems.

Once this initial set up is complete, a user will want to ensure that the new file system is always mounted on the systems where the analysis will be run. The user may choose to update the appropriate BPXPRMxx SYS1.PARMLIB members to ensure that the mount processing occurs.

SubTopic: The Data Privacy for Diagnostics Analyzer File System

The Data Privacy for Diagnostics Analyzer File System

The set-up job BLSDPJIN performs the following steps:

- Creates the file system
- Formats the file system
- Creates the home directory
- Mounts the file system to the home directory
- Runs the initialization shell script (blsdpdp.sh note)

The file system has a required folder structure. The following describes these sub-directories and their contents:

/<directory>/knowledgebase

This folder is used to store the ingested knowledge and user feedback.

/<directory>/knowledgebase/ingested/

This folder stores the ingested knowledge such as user provided directories and regular expressions, and is populated by the INGEST function.

/<directory>/knowledgebase/feedback/

This folder stores the processed user feedback and is populated by the FEEDBACK function.

/<directory>/configuration

This folder stores the configuration which is used for various operations carried out by Data Privacy for Diagnostics Analyzer. This folder will contain the following configuration files (which correspond to the ANALYZE, INGEST and EXTRACT modes of operations).

/<directory>/configuration/analysis_config.json

This file contains configuration about the sensitivity analysis to be carried out on dump. It allows customizing which built-in identifiers and ingested information that should be used for analysis. It also allows you to customize which combination of identifiers should be present together for data to be considered sensitive. See the z/OS MVS IPCS User's Guide for additional information about the analysis_config.json file including parameters and examples.

/<directory>/configuration/extract_config.json

This file contains configuration about identifiers which are to be extracted to a file. It allows the user to display the current pattern or dictionary associated with a built-in or customer identifier that is available for the ANALYZE function in determining which data is to be marked as sensitive by the Analyzer. See the z/OS MVS IPCS User's Guide for additional information about the extract_config.json file including parameters and examples.

/<directory>/configuration/ingestion_config.json

This file contains configuration about user provided data to be ingested. The ingested data is then available for use in future ANALYZE runs. See the z/OS MVS IPCS User's Guide for additional information about the ingestion_config.json file including parameters and examples.

/<directory>/reports

This folder is used to store reports generated by Data Privacy for Diagnostics Analyzer. A subdirectory is created for each dump on which Data Privacy for Diagnostics Analyzer ANALYZE processing is requested. The following folder structure is generated for each dump:

/<directory>/reports/<dump-name-1>/

This folder stores reports from each invocation of ANALYZE.

/<directory>/reports/<dump-name-1>/<timestamp-of-Data Privacy for Diagnostics Analyzer-ANALYZE-invocation>/

Stores reports of a single ANALYZE invocation. It contains the following files:

../concise_sensitive_report_<i>

This file is generated by each thread spawned by ANALYZE to process the dump.

../sensitive_token_log_<i>

This file is generated by each thread spawned by ANALYZE to containing all of the sensitive tokens identified in the dump. These files are generated by each thread spawned by the ANALYZE function if the value of the SENSITIVE REPORT field is Y on the IPCS ANALYZE panel or if the log_sensitive_tokens value is set to TRUE in the BLSJDPA JCL that invokes the ANALYZE function.

../non_sensitive_tokens

This file contains all the non-sensitive tokens identified in the dump along with their count. This file is generated when REPORT is requested after ANALYZE that requested token level redaction. Token level redaction can be requested by specifying the value N for the ALLOW PAGE LEVEL option on the ANALYZE IPCS panel or by specifying the value 2 for the analysis_mode option in the BLSJDPA JCL. This file can be modified to provide feedback about tokens which are incorrectly marked as non-sensitive.

../sensitive_tokens

This file contains all the sensitive tokens identified in the dump along with their count. This file is generated when REPORT is requested. This file can be modified to provide feedback about tokens which are incorrectly marked as sensitive for a subsequent FEEDBACK run.

z/OS MVS IPCS User's Guide (SA23-1384-00)

Topic: Using IPCS Functions

SubTopic: Using the IPCS Dialog

SubTopic: Using Data Privacy for Diagnostics

Data Privacy for Diagnostics Analyzer provides the capability to post process the following dump types taken on a z15 or later processor:

- SVC
- Stand-alone
- SLIP
- SYSMDUMP (from V2.5)
- Transaction (from V2.5)

Post processing is used to redact pages that have been tagged as being sensitive by the applications that created the pages, as well as untagged pages that will be scanned and detected as containing sensitive data per the Data Privacy for Diagnostics Analyzer, which requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0. This redacted version of the original dump is written to a new dump data set without modifying the original dump data set. Retain both dumps for as long as it takes to diagnose the reported problem.

Append Dump Directory records (BLSADDIR) are removed when generating a redacted stand-alone dump.

Additional processing is required for stand-alone dumps which contain captured dumps. If the captured dumps are required by vendors, the dumps must first be extracted (IPCS COPYCAPD) from the original stand-alone dump, then processed separately. Do NOT depend on captured dumps being available within a redacted stand-alone dump.

Note: A stand-alone dump can contain one or more SVC dumps that are captured in memory, but weren't written to a data set. It is recommended that you extract these SVC dumps using IPCS COPYCAPD, if captured on z15 or later processors, and post process them before sending them to IBM for further analysis to ensure that sensitive data is properly protected.

The following functions are being provided:

REDACT

You may redact any data tagged as sensitive=yes without further analysis.

Note: You cannot perform the ANALYZE function on a dump that has already been redacted via this process.

You can request this processing using either:

- IPCS option 5.6, specifying the ANALYZE function and BYPASS DP ANALYSIS=Y
- Use sample job SYS1.SAMPLIB(BLSJDPFD)

ANALYZE

Any pages tagged sensitive by the applications that own that data as well as any untagged pages detected as containing sensitive data.

Note: You cannot perform the ANALYZE function on a dump that has already been redacted via this process.

You can request this processing using either:

- IPCS option 5.6, specifying the ANALYZE function and BYPASS DP ANALYSIS=N
- Use sample job SYS1.SAMPLIB(BLSJDPA).

REPORT

You may create human readable reports for a dump that has been processed by the Data Privacy for Diagnostics Analyzer. These reports, once created, are in the <directory>/reports/<dump-name>/<run-number> directory in the file system used for DPA processing. You can request this processing using either:

- IPCS option 5.6, specifying the REPORT function.
- Use sample job SYS1.SAMPLIB(BLSJDPR).

FEEDBACK

You may provide feedback for a dump that has been processed by the Data Privacy for Diagnostics Analyzer. After looking through the reports and understanding the pages that have or have not been flagged as sensitive, you can provide feedback to help the Data Privacy for Diagnostics Analyzer improve its sensitive data detection. More information is covered on providing feedback later in this chapter. After updating configuration files and indicating what tagging can be improved, you can request this processing using either:

- IPCS option 5.6, specifying the FEEDBACK function
- Use sample job SYS1.SAMPLIB(BLSJDPF).

INGEST

You may ingest data to help the Data Privacy for Diagnostics Analyzer determine what sensitive data exists in your environment. Data can be ingested from dictionaries, databases or other sources. This data is added to the knowledge base information and will be used in future analysis runs. More information is covered on providing ingested data later in this chapter. After updating configuration files and indicating what tagging can be improved, you can request this processing using either:

- IPCS option 5.6, specifying the INGEST function
- Use sample job SYS1.SAMPLIB(BLSJDPI).

EXTRACT

You may extract any built-in or custom identifiers from the Analyzer to a file so that the user may see the exact criteria for determining the sensitivity of the data via the ANALYZE function. The output file will contain either the pattern or entire dictionary depending on the type of identifier to assist in ensuring that the Data Privacy for Diagnostics Analyzer is correctly marking data as sensitive or non-sensitive. More information is covered on extracting identifiers later in this chapter. After updating configuration files and indicating which identifiers can be written to a file, you can request this processing using either:

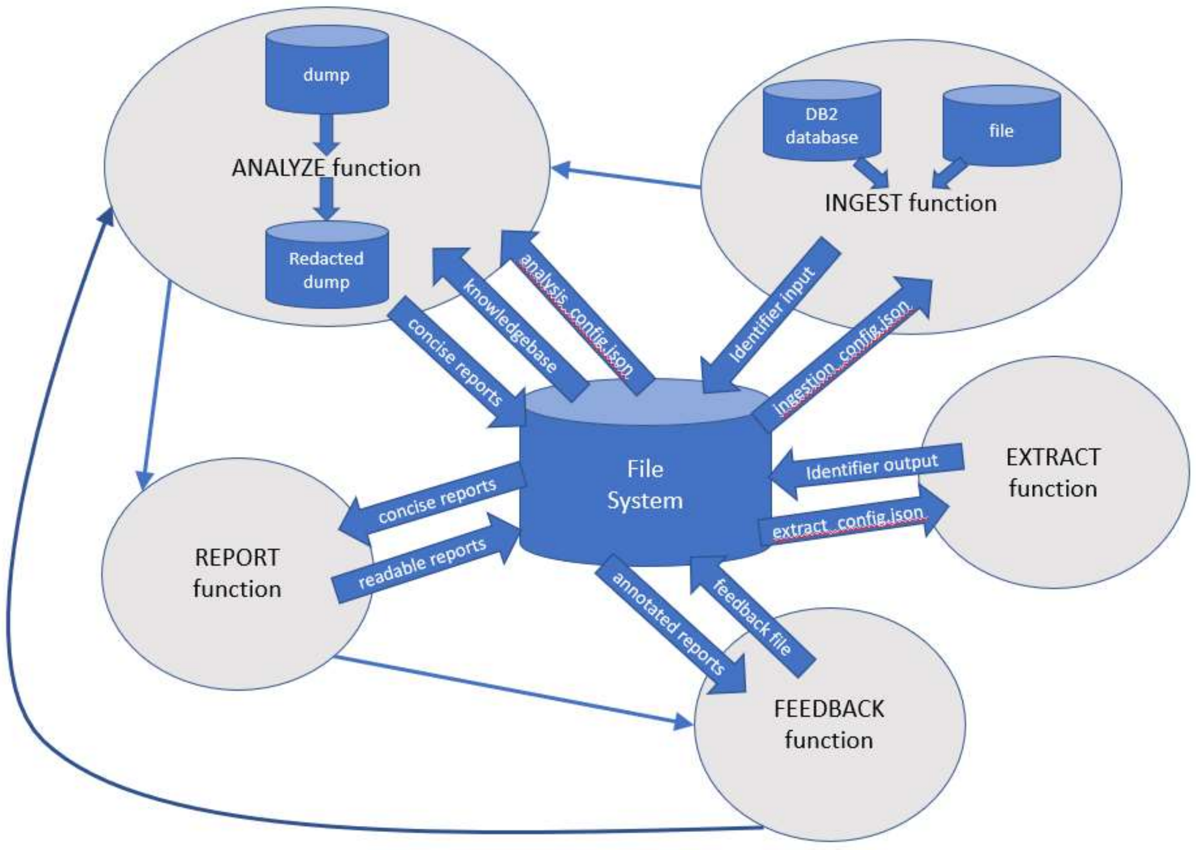
- IPCS option 5.6, specifying the EXTRACT function
- Use sample job SYS1.SAMPLIB(BLSJDPX).

Generally, you will want to start by performing the ANALYZE function on a dump. Remember that this function only works on dumps captured on a z15 or later processor. After creating the redacted version of the dump, you will want to check the dump to understand what has been redacted. Reports are available to help you understand why pages have been redacted. You can look at these reports to see if the data has been properly identified as sensitive. Some reports are written in concise form and must be formatted using the REPORT function. After running the REPORT function, you may want to give feedback to Data Privacy for Diagnostics Analyzer regarding some of the data that it either found as sensitive but was not actually sensitive, or feedback on data that was sensitive but not detected as sensitive. The FEEDBACK function allows you to perform this task. The cycle of ANALYZE / REPORT / FEEDBACK provides a way to train the Data Privacy for Diagnostics processing in order to produce dumps with the right level of redaction for your environment.

Another function that can be used is the INGEST function. This allows you to import data from databases and files, and lets you create custom information that can be used by the Data Privacy for Diagnostics Analyzer processing to help identify sensitive data.

In order to display the exact criteria that the ANALYZE function is using to determine data sensitivity, one could use the EXTRACT function to write out any built-in or custom identifiers to a file such that when that particular identifier is requested in the ANALYZE configuration, the user knows exactly which tokens or what pattern will be used to mark data as sensitive or non-sensitive.

Figure 1. Data Privacy for Diagnostics Usage Cycle



Using the Data Privacy for Diagnostics Analyzer Dialog within IPCS

When IPCS is used, panels are presented to allow you to specify parameters required for processing. The dialog generates appropriate JCL based on the parameters provided. If any data sets are required but not preallocated, the dialog attempts to dynamically allocate them. If dynamic allocation fails for any reason, you should be able to preallocate data sets using other mechanisms (such as ISPF option 3.2).

Note: Not all parameters are present on all IPCS panels for each function.

The parameters that are specified on the IPCS Data Privacy for Diagnostics Analyzer panels are:

DATA SET NAME
The input dump data set name. This option is equivalent to the input_dataset parameter in the JCL submitted to perform the requested function.

NEW DATA SET NAME
The output (redacted) dump data set name. This option is equivalent to the output-dump-dataset field in the JCL submitted to perform the ANALYZE function.

TEMP DATA SET/PAT
Temporary data set names can either be a specific name or a data set name pattern. See the help panels for more information on patterns. This option is equivalent to the output_dataset or output_dataset_prefix parameters in the JCL submitted to perform the requested function.

BYPASS DP ANALYSIS
Allows you to submit a job that will either perform analysis (N) or skip analysis (Y). If N is specified, the Data Privacy for Diagnostics Analyzer step will scan the input data set looking for additional sensitive data in addition to data identified by the applications that allocated the storage marked as sensitive. If found, either token-level or page level redaction will be performed based on the Allow Page Level specification. If Y is specified, this step will be bypassed. The output data set identified by the NEW DATA SET NAME field will only have data removed that was identified by the applications that allocated the storage marked as sensitive.

REDACTION STRING
If you are not allowing page level redaction, this redaction string is used to overlay data determined to be sensitive in the output dump. You may leave this field blank to overlay the token with X or specify a string. When longer strings are detected in the pages, the string is used in a repeated fashion. If shorter strings are found, only a portion of the redaction string may be used. This option is equivalent to the redaction_string parameter in the JCL submitted to perform the requested function.

NUMBER OF THREADS
For ANALYZE requests, large dumps may be processed faster by using multi-threading. You may specify 1 to 8 for the number of threads. Each thread requested will process a portion of the input dump, reducing the elapsed time it takes to process the entire dump, however, it may also increase the simultaneous amount of resources required to process the request. This option is equivalent to the thread_count parameter in the JCL submitted to perform the requested function.

ALLOW PAGE LEVEL
If Y is specified, known as fast-analysis mode or page-level redaction, the entire page of storage is redacted when any sensitive data is detected. Page-level redaction may allow the analysis processing to run faster since processing will stop at the first sensitive string in a page is found, however, it is possible that allowing page-level

redaction may cause diagnostic data to be lost. If you find this to be true, set the value to N, known as detailed analysis mode or token-level redaction, so that data that is determined to be sensitive will be overlaid using only the redaction string. The default value is N or token-level redaction.

SENSITIVE REPORT

If Y is specified, reports are generated in <directory>/reports/<dump-name>/<run-number>/sensitive_token_log_*n* where n is the thread number. There will be a file per thread requested. For each string detected, data is written to these files to help you understand what has been redacted and why. Based on this information, you may decide to include or exclude types of data. When the REPORT function is requested, it will consolidate these sensitive_token_log_*n* files into a human-readable file named sensitive_tokens.

DPfD HOME DIR

Specify the path where the Data Privacy for Diagnostics Analyzer home directory is configured, <directory> as previously described. Do not include the trailing '/' when specifying this path.

JAVA HOME DIR

Specify the path where java is installed. This will be used in the batch job's STDENV set up file to create the proper environment for the java processing to run in. Do not include the trailing '/' when specifying this path. Data Privacy for Diagnostics requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0.

JAVA OPTIONS

You may provide whatever java options are desired. For example, you may need to specify a minimum and maximum heap size for the JVM to successfully run a multi-threaded DPfD ANALYZE request. Requesting additional threads and/or including additional identifiers will increase the size of the heap for the JVM, so use the –Xms and –Xmx options to adjust the minimum and maximum heap size. For more information on JVM Command-Line Options, see the topic OpenJ9 command-line options in IBM SDK, Java Technology Edition 8.0.0 (https://www.ibm.com/support/knowledgecenter/en/SSYKE2_8.0.0/openj9/cmdline_specifying/index.html)

Note: Data Privacy for Diagnostics requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0.

JZOS LOAD MODULE

The dialog uses the JZOS Batch Launcher in the JCL that is submitted. You should determine the correct level of JZOS installed on your system and provide the name of the appropriate load module in this parameter. Data Privacy for Diagnostics requires a minimum of IBM 64-bit SDK for z/OS Java Technology Edition version 8.0, thus the 64-bit version 8 load module for JZOS Batch Launcher is JVMLDM86. For additional information, see the JZOS Batch Launcher and Toolkit Installation and Users Guide.

(https://www.ibm.com/support/knowledgecenter/en/SSYKE2_8.0.0/com.ibm.java.zsecurity.80.doc/zsecurity-component/jzos.html)

MIGLAB DATASET

A sort E35 exit is used to remove pages flagged as sensitive. This function is provided in module BLSRTE35 which is shipped in SYS1.MIGLIB. Should you need to override where this exit can be loaded from, provide the name of the MIGLIB that contains the load module you wish to run.

TEMP ALLOC PARMS

If your environment requires specific allocation parameters for dump data sets, you may supply any allocation parameters that will ensure the data set is properly allocated. For example, supplying DATACLAS and STORCLAS keywords may be necessary to locate the correct storage pool and attributes.

Note: Do NOT specify RECFM, DSORG, LRECL, BLKSIZE, SPACE and TRACK as they are used to create some of the interim data sets. If you need to use one of those allocation parameters, request the ANALYZE function via the JCL instead of through IPCS.

EDIT CONFIG FILE?

If Y, allows the user to edit the configuration file pertaining to the function requested (analysis_config.json for ANALYZE or ingestion_config.json for INGEST or extract_config.json for EXTRACT) prior to submitting the JCL to perform the requested function. Default is N. See the analysis_config.json, extra_config.json and ingestion_config.json sections for additional information.

RUN NUMBER

From the ANALYZE step, a run number was generated and can be found in the job output which can be specified for this parameter when the function requested is REPORT or FEEDBACK. If a run number is NOT specified, the most recent ANALYZE run for the input dump is used.

DB2 JDBC PATH

For the INGEST function, if using database as the source in the ingestion_config.json file, this field is needed to specify the path for the DB2 JDBC Driver and License JAR files. Do not include the trailing '/' when specifying this path.

Topic: Using IPCS Functions
SubTopic: Using the IPCS Dialog
SubTopic: Using Data Privacy for Diagnostics
SubTopic: Requesting an ANALYZE run

Requesting an ANALYZE run

ANALYZE is the function that will look for sensitive data in dump records and flag those records for redaction, or even overlay that sensitive data with a redaction string if token-level redaction is requested. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPA JCL in SYS1.SAMPLIB) , two important configuration files are used:

- Runtime configuration file
    - This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPA JCL.
- analysis_config.json file
    - This file provides additional detail on what to include or exclude while looking for sensitive data in dump records. It is located in the <directory>/configuration/ directory in the file system. You may either use the EDIT CONFIG FILE option Y in the ANALYZE IPCS panel to edit this file if you want to change it, or you may directly edit it using an editor that you are familiar with.

The Runtime configuration is a json file that is built by the dialog using the parameters supplied on the panel, or can also be supplied as a file or an instream data set in JCL if you use JCL to submit the job. The runtime configuration file parameters (hand-coded in the BLSJDPA JCL) are:

"input_dataset"
Specifies the location of the input dump. You must specify a data set name as follows: "//'<dump-dataset-name>'"
If using the ANALYZE IPCS panel, this parameter is populated by the DATA SET NAME field.

"thread_count"
This specifies the number of worker threads to be spawned. The total number of threads is one more than this (there is one monitor thread). If omitted, the default value is 4 for the JCL interface. Valid values are 1-8. If using the ANALYZE IPCS panel, this parameter is populated by the NUMBER OF THREADS field.

"record_count"
Estimated number of records in the input dump. If set to 0 or omitted, the Data Privacy for Diagnostics Analyzer will count the actual number of records. This is used when

multiple threads are requested to ensure that the records are evenly split by the requested number of threads. This parameter is not available in IPCS panel interface.

"output_dataset_prefix" or "output_dataset"

Specifies either the prefix to use for each thread's output dump data, or the list of dataset(s). Either data set names or DD names may be specified on this parameter. For example, "output_dataset_prefix":"//’SYS1.DUMP.D190926.T132348’" indicates the prefix that will be used by each thread. Files will either be dynamically allocated or may be pre-allocated as SYS1.DUMP.D190926.T132348.F1, SYS1.DUMP.D190926.T132348.F2, SYS1.DUMP.D190926.T132348.F3, etc., one per thread. When using the BLSJDPA JCL, you may also specify DDNAMEs as prefixes. For example, you may specify "output_dataset_prefix":"//DD:ANLZO" and specify DD statements for //ANLZOF1, //ANLZOF2, etc, for each thread's output. Alternatively, you may use the "output_dataset" method of supplying a list of datasets. For example, you may specify "output_dataset":["//’SYS1.DUMP.D190926.T132348.F1’","//’SYS1.DUMP.D190926.T132348.F2’"]. If you are using the dialog to initiate the job, you may also specify a pattern to be used. In this case, the pattern may contain a single "%" character which will cause the dialog to generate data set names with thread numbers substituted in that position in the data set name. For example, you may specify 'SYS1.DUMP.D190926.T132348.F%' as the data set name pattern on the panel and the dialog would generate "output_dataset":["//’SYS1.DUMP.D190926.T132348.F1’","//’SYS1.DUMP.D190926.T132348.F2’"] as the parameters.

If using the ANALYZE IPCS panel, this parameter is populated by the TEMP DATA SET/PAT field.

"redaction_string"

String to use for redaction for pages being analyzed using detailed analysis. When a redaction_string isn’t specified and detailed analysis is specified, sensitive data will be replaced with X. When longer strings are detected in the pages, the string is used in a repeated fashion. If shorter strings are found, only a portion of the redaction string may be used. If using the ANALYZE IPCS panel, this parameter is populated by the REDACTION STRING field.

"analysis_mode"

Specifies the analysis mode for detecting sensitive data. Valid values are 1 for Page-Level Redaction and 2 for Token-Level Redaction. In Page-Level Redaction, the entire page is marked as sensitive as soon as first sensitive token is identified in the page. In Token-Level Redaction, each sensitive token is identified independently and overlaid with the redaction_string. Note that when 1 is specified, some pages may be analyzed using Token-Level Redaction. If using the ANALYZE IPCS panel, this parameter is populated by the ALLOW PAGE LEVEL field, with Y being equivalent to 1 (Page-Level Redaction) and N being equivalent to 2 (Token-Level Redaction).

"log_sensitive_tokens"
Specifies if the sensitive token log for each file should be generated. When set to TRUE, a sensitive token log for each thread should be generated in the <directory>/reports folder. Valid values are TRUE and FALSE. If using the ANALYZE IPCS panel, this parameter is populated by the SENSITIVE REPORT field, with Y being equivalent to TRUE and N being equivalent to FALSE. NOTE: by specifying TRUE, an additional file will be generated on each ANALYZE function request, thus the Data Privacy for Diagnostics Analyzer home directory will fill up more quickly.
"dpfd_home"
Specifies the Data Privacy for Diagnostics Analyzer home directory. If using the ANALYZE IPCS panel, this parameter is populated by the DPfD HOME DIR field.
"character_set"
Specified the character set that should be used for decoding of the input dump. The default value is "Cp1047". This parameter is not available in IPCS panel interface. Valid values are Cp1047 and US-ASCII.

The analysis_config.json file:
As your experience with this ANALYZE processing matures, this file will likely become stable until major changes in data occur in your environment. You will likely start with the default file supplied with the product. As your usage evolves, you may decide to exclude or include certain built-in identifiers, use custom identifiers or add dependent identifiers. This file is in JSON format containing keyword-value pairs and arrays used to specify parameters to the ANALYZE function.
Parameter Descriptions:
"built_in_identifiers_include"
This specifies the built-in identifiers which should be used to detect sensitive tokens. Values that can be supplied here are listed in the table below. If nothing is specified, no identifiers are included. These built-in identifiers are specified in a comma-separated list with quotes around the identifier

```
"built_in_identifiers_include" : [
"Identifier1",
"Identifier2",
...
"IdentifierN"
],
```

See Figure 1 analysis_config.json example for an example of the full file.

"built_in_identifiers_exclude"
This specifies the built-in identifiers which should not be used to detect type of tokens. Values that can be supplied here are listed in the table below. If nothing is specified, no identifiers are excluded. These built-in identifiers are specified in a comma-

separated list with quotes around the identifier as shown in Figure 1 analysis_config.json example.

"custom_identifiers"
This specifies any custom identifiers which should be used to detect sensitive data. Custom identifiers can be ingested using the INGEST function and can be in the form of dictionaries or patterns. Arrays of multiple identifiers may be specified with the attributes of a single custom identifier enclosed in { } as described below and the identifiers themselves are separated by commas as shown in Figure 1 analysis_config.json example. Each identifier has the following fields:

"format"
: Valid value is "custom". "custom" indicates that a previously ingested dictionary or set of patterns is specified.

"inputfilename"
: File name containing the identifier data. The Data Privacy for Diagnostics Analyzer looks for the file in /<directory>/knowledgebase/ingested/.

"entitytype"
: Specifies a name for the identifier. This is used when it is part of dependent identifier. If the entity name was provided when the file was ingested (for "custom" format), this field can be skipped. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.

"description"
: Specifies description of the identifier. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.

"dependent_identifiers"
This specifies a set of identifiers which are sensitive only when they all occur in a page. Here, you specify an array where multiple dependent identifiers can be specified with the attributes of a single dependency enclosed in { } as described below and the dependencies themselves are separated by commas as shown in Figure 1 analysis_config.json example. When an identifier is made part of dependent_identifier, it stops being an independent identifier. Note that all of the identifiers specified in a dependent_identifier should be present in built_in_identifiers_include or in custom_identifiers; otherwise, this will never be detected. Each dependent identifier has the following fields:

"name"
: Specifies name assigned to this dependent identifier.

"identifiers"

Specifies the set of identifiers belonging to this dependent identifier. These identifiers are specified in a comma-separated list with quotes around the identifier as shown in Figure 1 analysis_config.json example.

“built_in_ns_identifiers_include”

This specifies the built-in identifiers which should be used to detect that a token is non-sensitive. These built-in identifiers are specified in a comma-separated list with quotes around the identifier as shown in Figure 1 analysis_config.json example.

“built_in_ns_identifiers_exclude”

This specifies the built-in identifiers which should not be used to detect that a token is non-sensitive. These built-in identifiers are specified in a comma-separated list with quotes around the identifier as shown in Figure 1 analysis_config.json example.

“custom_ns_identifiers”

This specifies any custom identifiers which should be used to identify tokens as non-sensitive data. Custom identifiers can be ingested using the INGEST function and can be in the form of dictionaries, patterns with the attributes of a single custom identifier enclosed in { } as described below and the identifiers themselves are separated by commas as shown in Figure 1 analysis_config.json example. Each identifier has the following fields:

"format"

Valid value is "custom". "custom" indicates that a previously ingested dictionary or set of patterns is specified.

"inputfilename"

File name containing the identifier data. The Data Privacy for Diagnostics Analyzer looks for the file in /<directory>/knowledgebase/ingested/.

"entitytype"

Specifies a name for the identifier. This is used when it is part of dependent identifier. If the entity name was provided when the file was ingested (for "custom" format), then this field can be skipped. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.

"description"

Specifies description of the identifier. If values are provided both during ingestion and in analysis_config.json, the value provided in analysis_config takes precedence. If no value is provided either during ingestion or in analysis_config, a custom value is chosen.

“printable_characters”
This field specifies set of printable characters which will be used for analysis of dumps. When analyzing the dump, only these characters are used to construct the tokens that will be parsed. Default value is "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890`~!@#$%^&*()-=_+ {}[];:'\",<.>/?\\|\n "

Any incorrect information in analysis_config.json ignored. Identifiers specified in the built_in_identifiers_exclude list take precedence over identifiers specified in the built_in_identifiers_include list.

Figure 1. analysis_config.json example
Note: The custom_identifiers and custom_ns_identifiers are assumed to have been created via the INGEST function in this example.
```
{
"built_in_identifiers_include" : [
"Month",
"FullName",
"Credit Card Type",
"Credit Card Number",
"Email",
"Zipcode",
"Day"
],
"built_in_identifiers_exclude": [
"Year",
"Date Time"
],
"custom_identifiers": [
{
"inputfilename":"acctnum.bin",
"entitytype" : "Account Number",
"description" : "List of account numbers",
"format" : "custom"
},
{
"inputfilename" : "policynum.bin”,
"entitytype" : "PolicyNumber",
"description" : "List of policy numbers",
"format" : "custom"
}
],
"dependent_identifiers": [
{
"name": "Full Person",
"identifiers": [ "FullName", "Zipcode", "Email" ]
},
{
"name": "Card",
"identifiers": [ "Credit Card Type", "Credit Card Number"]
}
],
"built_in_ns_identifiers_include" : [
"ModuleName"
],
"built_in_ns_identifiers_exclude": [
],
"custom_ns_identifiers": [
{
"inputfilename":"branch.bin",
"entitytype" : "Branch Name",
"description" : "List of branch locations",
"format" : "custom"
},
{
"inputfilename" : "zone.bin”,
"entitytype" : "Zone",
"description" : "List of zones",
"format" : "custom"
}
]
}
```

The built_in_identifiers supplied with Data Privacy for Diagnostics Analyzer are (by their very nature are not necessarily all inclusive of the particular topic, and may not be changed over time should there be changes to the actual data set of said topic):

Identifier (not case sensitive)	Description
age	String patterns related to age. Examples: “10 years old” “4 months old” “dob: 1-2-1999" **Note:** These string patterns will be considered sensitive, but just the number 10 (in the first example) will not be considered sensitive by itself.
continent	Dictionary containing the names of continents.
country	Dictionary containing the names of countries.

county	Dictionary containing the names of all the counties in US. In the United States of America, an administrative or political subdivision of a state is a county
credit card type	Credit card identification dictionary. Cards detected are VISA, Mastercard, AMEX, Diners Club, Discover and JCB
credit card number	Credit card pattern identification. Cards patterns detected are VISA, Mastercard, AMEX, Diners Club, Discover and JCB
date time	Date and Time pattern identification
day	Dictionary containing the names of the days of the week.
dependent	Dictionary containing the names of types of dependents, such as daughter, son, etc.
email	Email address pattern.
eu nin	National Identification Number patterns for various EU countries.
FullName	A first name and last name pair dictionary, the combination of which is from popular names in the US census. NOTE: This identifier will detect a 2-word combination of first name and last name separated by a delimiter of a comma and/or spaces in either order. This identifier will NOT detect a name that contains any middle name/initial nor hyphenated names nor names that contain apostrophes.
gender	Dictionary containing the genders Male and Female.
iban	International Bank Account Number (IBAN) pattern
icdv9	International Classification of Diseases 9th Revision (ICDv9) identification dictionary.
icdv10	International Classification of Diseases 10th Revision (ICDv10) identification dictionary.

imei	International Mobile Equipment Identity (IMEI) identification dictionary. NOTE: This identifier only detects 15-digit IMEI tokens.
imsi	International Mobile Subscriber Identity (IMSI) Identification dictionary.
in aadhaar card number	Aadhaar identification number for residents or passport holders of India.
in PAN card number	Permanent Account Number pattern issued by Indian Income Tax Department
international phone number	International phone number identification pattern.
ip address	IP address identification pattern. Supports both IPv4 and IPv6 addresses
latitude longitude	Latitude/longitude identification pattern. Supports GPS and DMS coordinates formats, Ex: 12:30'23.256547S 12:30'23.256547E, N90.00.00 E180.00.00
mac address	MAC Address Identification pattern.
marital status	Marital status identifier dictionary.
medical name	Medical Name identification pattern. Example: John Doe MD
medical record number	Medical Record Number identification pattern, for example, MRN: CLM-00000056055, Medical Record Number: 1234asds
month	Month Name identification dictionary.
occupation	Occupation identification dictionary.
PO box	Identifies post office box numbers, Ex: P.O. BOX 334, POBOX 14321412
raceOrEthnicity	Dictionary identification of ethnic groups
religion	Dictionary containing major religions.
street types	Street Type identification, for example, tokens containing "st."
uk nin	National Insurance Number pattern.
us address	Identifies US-centric address patterns like “800 Theatre Court Garden City, NY 11530”. This just checks the format but

	does not validate city and state/zip in the address.
us phone number	US specific phone/fax/pager identifier pattern.
us ssn	US Social Security Number pattern
us states	US State Name identification
vehicle identification number	Vehicle identification number identification dictionary. Supports world manufacturer identification dictionary.
year	Year of Birth Identification, Any number between 0 and current year.
zipcode	Valid US zip code identifier dictionary.

Note: The following identifiers are no longer valid as of OA61591: Animal, ATC, Hospital Name, Phone Number, US SWIFT Code

The default analysis_config.json file will include the list of included identifiers and excluded identifiers.

Identifier checking is done using the following order:
1. User feedback indicates that a token is sensitive
2. User feedback indicates that a token is non-sensitive
3. module name check
4. Non sensitive checks (built-in + custom)
5. sensitive checks (built-in + custom)

NOTE: If the identifier is added into both the _include list and the _exclude list for either sensitive or non-sensitive checks, it will be treated as excluded.

Topic: Using IPCS Functions
SubTopic: Using the IPCS Dialog
SubTopic: Using Data Privacy for Diagnostics
SubTopic: Requesting a REPORT run

Requesting a REPORT run

REPORT is the function that will format concise reports in the file system into human-readable reports for the requested dump, and serves as the input to the FEEDBACK function. By default, the reports for the last ANALYZE run will be formatted. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPR JCL), the runtime configuration file is used to specify options for processing. This file is either built by the

dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPR JCL. The runtime configuration file parameters (built by dialog or hand-coded in the BLSJDPR JCL) are:
"input_dataset"
Specifies the location of the input dump. You must specify a data set name as follows: "//'<dump-dataset-name>'". During ANALYZE processing, the dump name was used as a directory and will be used by REPORT processing to locate the files produced during ANALYZE processing.
"dpfd_home"
Specifies the Data Privacy for Diagnostics Analyzer home directory.
"run_number"
Specifies the run number. If you omit the run_number option, it will use the most recent ANALYZE run.
As you validate how accurate the ANALYZE processing was in detecting sensitive data in your dumps, you may need to provide feedback to help the ANALYZE processing. In order to provide this feedback, you need to run the REPORT processing prior to the FEEDBACK function. The outputs of the report processing are human-readable files that can be edited should you need to provide FEEDBACK for future ANALYZE attempts. The files produced from the REPORT function are:
/<directory>/reports/<dump-name>/<run-number>/sensitive_tokens
This file contains the list of each token that was found to be sensitive data.
/<directory>/reports/<dump-name>/<run-number>/non_sensitive_tokens
this file, if requested, contains the list of each token that was found to be non-sensitive data. Note that this file is only produced if Page-Level Redaction was requested during the ANALYZE processing.

Topic: Using IPCS Functions
SubTopic: Using the IPCS Dialog
SubTopic: Using Data Privacy for Diagnostics
SubTopic: Requesting a FEEDBACK run

Requesting a FEEDBACK run
After running the ANALYZE processing, followed by the REPORT processing, you may want to provide feedback to enhance the accuracy of future ANALYZE functions in detecting the appropriate sensitive data for your environment. You can provide feedback to indicate the following:

- Tokens found to be sensitive are not actually sensitive
- Tokens not found to be sensitive are actually sensitive

To do this, you must edit the reports generated from the REPORT processing. These reports will be found in the /<directory>/reports/<dump-name>/<run-number> directory.

- In the sensitive_tokens file, change the “Is_Analysis_Correct” field from “Y” to “N” for any token that should not be considered sensitive.
- In the non_sensitive_tokens file, change the “Is_Analysis_Correct” field from “Y” to “N” for any token that should be considered sensitive.

Afterwards, the FEEDBACK function can be requested. Regardless of how the job is initiated (via IPCS option 5.6 or via the BLSJDPF JCL), the runtime configuration file is used to specify options for processing. This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPF JCL. The runtime configuration file parameters (built by dialog or hand-coded in the BLSJDPF JCL) are:

“input_dataset”

Specifies the location of the input dump. You must specify a data set name as follows: “//’<dump-dataset-name>’”. During ANALYZE processing, the dump name was used as a directory and will be used by FEEDBACK processing to locate the files produced during ANALYZE processing.

“dpfd_home”

Specifies the Data Privacy for Diagnostics Analyzer home directory.

“run_number”

Specifies the run number. If you omit the run_number option, it will use the most recent ANALYZE run. Ensure that the run_number is the same directory in which the edited reports are contained.

During the FEEDBACK operation, the Data Privacy for Diagnostics Analyzer reads these edited reports and updates the <directory>/knowledgebase/feedback/feedback.bin file, which will be used for future ANALYZE runs.

NOTE: If the redaction string is marked as a sensitive token via FEEDBACK, it will not be treated as a sensitive token.

Topic: Using IPCS Functions
SubTopic: Using the IPCS Dialog
SubTopic: Using Data Privacy for Diagnostics
SubTopic: Requesting an INGEST run

Requesting an INGEST run

You may want to customize the detection of sensitive data that is unique to your environment. This can be achieved by the INGEST function, which will help the Data Privacy for Diagnostics Analyzer to detect the sensitive data in future analysis. You can initiate the INGEST function from either IPCS option 5.6 or the BLSJDPI JCL. Either way, it will use the following files:

Runtime configuration file

This file is either built by the dialog using the supplied parameters, or is specified via an in-stream DD statement in the BLSJDPI JCL.

ingestion_config.json file

This file provides detail on what the Analyzer which then can be used while analyzing diagnostic data. It is located in the <directory>/configuration/ directory in the Data Privacy for Diagnostics file system. You may either use the EDIT CONFIG FILE option Y in the INGEST IPCS panel to edit this file if you want to change it, or you may directly edit it using an editor that you are familiar with.

Regardless of how the job is initiated, the runtime configuration file is used to specify options for processing. The runtime configuration file parameters (Built by dialog or hand-coded in the BLSJDPI JCL) are:

"dpfd_home"

Specifies the Data Privacy for Diagnostics Analyzer home directory.

The ingestion_config.json file contains information about the identifiers to be built as sensitive or non-sensitive tokens based on the options specified. INGEST will generate a file in the <directory>/knowledgebase/ingested directory, which can be subsequently specified in the analysis_config.json file to add the ingested identifier as a custom identifier for sensitive data detection during the ANALYZE function. This data can be ingested from dictionaries, databases or other sources.

“outputfilename”

Specifies the name of the file to be stored in the <directory>/knowledgebase/ingested directory after successful INGEST run. This can be specified in the analysis_config.json file as the inputfilename under the custom_identifiers option for use in future ANALYZE requests.

“entitytype”

Specifies the name of the identifier. This is used when user does not provide one in the analysis_config.json file in the custom_identifiers option under entitytype.

“description”

Specifies a description of the identifier. This is used in future REPORTs when user does not provide one in the analysis_config.json file in the custom_identifiers option under description.

"inputtype"

Specifies the type of input to INGEST. Valid values are:

"pattern"

Allows the specification of a Java Regex (or Java Regular Expression) as a pattern for matching strings.

"dictionary"

Allows specification of exact tokens to be matched by the Data Privacy for Diagnostics Analyzer.

"inputsource"

Specifies the source of the input data to be ingested. Valid values are:

"file"

Indicates that the source is a file with the location specified on the “inputfilename” option.

"inline"

Indicates that the source is inline data specified in the ingestion_config.json file.

“database”

Indicates that the source is a database as specified in the “database” and associated options. This option is only valid when “inputtype” of ”dictionary” is specified.

Note: If you chose this option, you MUST update the DB2 JDBC PATH, by either entering the full path to the DB2 JDBC Driver and License JARs in the DB2 JDBC PATH field in IPCS option 5.6 or update the CLASSPATH section of the STDENV DD in the BLSJDPI JCL to include the full path.

“inputfilename”

Specifies the path and name of the file which contains the data to be used during the INGEST function. This option is only valid when “inputsource”:”file” is specified. The format for specifying a dictionary or pattern in a file is 1 entry per line as such:
`value1value2value3`

“inlinedata”

Specifies the data to be used during the INGEST function. This option is only valid when “inputsource”:”inline” is specified. The format for specifying inline data is:
"inlinedata" : ["value1", "value2", "value3"]

“database”

Specifies the type of database to be used as an input source. This option is only valid when “inputsource” of ”database” is specified. Valid values are:

“DB2"

Indicates the database is DB2 installed on non-system Z platform.

“DB2zOS”

Indicates the database is DB2 installed on system-Z.

“DB2zOSptkt”

Indicates the database is DB2 to be connected via pass ticket. This is the default value.

“databasehost”

Specifies the domain name or IP address where the database is hosted. This option is only valid when “inputsource” of ”database” is specified. The format for specifying this option is: “databasehost”:”<url>”
“databaseport”
Specifies the port number that identifies the DB2 subsystem. This option is only valid when “inputsource”:”database” is specified.
“databaseusername”
Specifies the user ID used to connect to the DB2 database. This option is only valid when “inputsource”:”database” is specified.
“databasepassword”
Specifies the password for the user ID used to connect to the DB2 database. This option is only valid when “inputsource”:”database” is specified.
“databasename”
Specifies the name of the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
“databaseschema”
Specifies the schema in the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
“databasetablename”
Specifies the name of the table in the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
“databasecolumnname”
Specifies the name of the column in the database containing the data to be ingested. This option is only valid when “inputsource”:”database” is specified.
Any options specified in the ingestion_config.json file which are not valid with the specified input source will be ignored. Only 1 value per option will be processed, except for the inline data option which may be specified in a list form. After the INGEST request is completed, the newly created identifier must be specified in the custom_identifiers options of the analysis_config.json file in order to be considered for determining sensitive data for the subsequent ANALYZE requests.
Note: Only 1 custom identifier may be specified per INGEST request.

The following are examples of ingestion_config.json files.
To create a new sensitive identifier called account that will detect the account number of customers in a dump, a pattern can be used if a certain format for the account numbers is known. For example, an account number beginning with 2 alphabetical characters followed by 8 numeric digits. The following is an inline pattern using Java Regex.

```
{
"inputtype":"pattern",
"inputsource":"inline",
"entitytype":"account",
"description":"a pattern to determine account: 2 characters, 8 digits",
"outputfilename":"accts.bin",
"inlinedata":["\\D{2}\\d{8}"]
}
```
Note: That \ is an escape character in Java strings, which requires you to use \\ to define a single \ in order to use meta Regex characters like \D (non-digit) and \d (digit).
If you have a file that contains all of the account numbers for your customers, you can alternatively provide a file dictionary to create your custom identifier:
```
{
"inputtype":"dictionary",
"inputsource":"file",
"entitytype":"accounts",
"description":"a list of our customer accounts",
"outputfilename":"accts.bin",
"inputfilename":”/u/ibmuser/accounts.txt”
}
```
You may also INGEST a column from a DB2 database as a dictionary that can be used to identify sensitive data:
```
{
"inputtype":"dictionary",
"inputsource":"database",
"entitytype":"accounts",
"description":"a list of our customer accounts",
"outputfilename":"accts.bin",
"database":"db2zos",
"databasehost":"db2host.pok.ibm.com",
"databaseport":"446",
"databaseusername":"db2user",
"databasepassword":"Bot27tle",
"databaselocation":"DBX5LOC1",
"databaseschema":"dsn8910",
"databasetablename":"DBO",
"databasecolumnname":"ACCOUNTNUMBER"
}
```
After the INGEST process has been completed, in order to use your newly created customer identifier in the subsequent ANALYZE request, you must amend your analysis_config.json file to add the accounts to the custom identifier as such:
```
"custom_identifiers":
[
      {
      "inputfilename" : "accts.bin",
      "entitytype" : "accounts",
      "description" : "Customer Accounts",
      "format" : "custom"
      }
]
```

z/OS MVS IPCS Commands (SA23-1382-00)

Topic: IPCS CLISTs and REXX EXECs

SubTopic: BLSXREDR REXX EXEC – Report Pages Marked as Sensitive

BLSXREDR REXX EXEC — report pages marked as sensitive

Reports the pages marked sensitive.

Use the BLSXREDR EXEC to generate a report about the parameters used to perform analysis via Data Privacy for Diagnostics to produce the redacted dump (if the ANALYZE function was used via the IPCS panels, but not via the BLSJDPA JCL) as well as ranges of pages in a dump which were tagged with the SENSITIVE=YES attribute (if the dump is not post-processed) or redacted by the Data Privacy for Diagnostics Analyzer and being tagged with the SENSITIVE=YES attribute (if the dump is post-processed).

Syntax

```
For non-IPCS environment:
%BLSXREDR <dump_dsn> [A <asid>] [DETAILS]

For IPCS environment:
BLSXREDR [A <asid>]
-OR-
BLSXREDR [ADDR <address>]
```

Parameters

<dump_dsn>

- Specify the name of a post-processed dump data set.
  - **Note:** If using BLSXREDR within IPCS, this parameter cannot be specified, and the currently ACTIVE dump data set will be used.

A <asid>

- [Optional] Specify an address space identifier number in hex as a filter to reduce the output to contain only addresses for a specific address space. If omitted, all ASIDs will be displayed.
  - **Note:** ASID can be used instead of A. This parameter cannot be specified with the ADDR parameter.

DETAILS

- [Optional] When specified, displays all pages that were tagged as SENSITIVE=NO in addition to pages which were partially redacted by the Data Privacy for Diagnostics Analyzer (for post-processed dumps) or marked redactable via being tagged SENSITIVE=YES (for not post-processed dumps). This does NOT display page ranges in which entire pages were removed from the post-processed dump due to being tagged as SENSITIVE=YES or via the Data Privacy for Diagnostics Analyzer when Allow Page Level was set to Y during the ANALYZE function for post-processed dumps.
  - **Note:** If using BLSXREDR within IPCS, this parameter cannot be specified.

ADDR <address>

- [Optional] When specified, allows the caller to filter the ranges presented as output to include the passed address. All ASIDs and areas will be examined for a range encompassing the passed address.
    - **Note:** This parameter can only be specified if using BLSXREDR within IPCS and cannot be specified with the A/ASID nor the DETAILS parameters.